Multi-Texture Image Segmentation

L M Linnett B.Sc., M.Sc.

Thesis submitted
for the
Degree of Doctor of Philosophy

Heriot-Watt University

Department of Electrical and Electronic Engineering

March 1991

TABLE OF CONTENTS

LIST OF FIGURES.

LIST OF TABLES.

ACKNOWLEDGEMENTS

The author wishes to thank Professor G.T. Russell whose encouragement, stimulating discussion and advice, are especially appreciated.

I would like to thank the staff of the Department of Electrical and Electronic Engineering at Heriot-Watt University.

My thanks also go to certain individuals in the non-academic world, who have supported the work both financially and with their enthusiasm, in particular, Mr.D.N.Langhorne, Mr.J.Wickenden and Dr.A.Burgess of the Admiralty Research Establishment (ARE Portland), and Mr. G.A. Whittaker of Honeywell (USA).

Finally I would like to thank my family for their forbearance and also my three cats, Tess, Penny and Pouncer, for their lessons in patience and perseverance.

ABSTRACT

Visual perception of images is closely related to the recognition of the different texture areas within an image. Identifying the boundaries of these regions is an important step in image analysis and image understanding. This thesis presents supervised and unsupervised methods which allow an efficient segmentation of the texture regions within multi-texture images.

The features used by the methods are based on a measure of the fractal dimension of surfaces in several directions, which allows the transformation of the image into a set of feature images, however no direct measurement of the fractal dimension is made. Using this set of features, supervised and unsupervised, statistical processing schemes are presented which produce low classification error rates. Natural texture images are examined with particular application to the analysis of sonar images of the seabed.

A number of processes based on fractal models for texture synthesis are also presented. These are used to produce realistic images of natural textures, again with particular reference to sonar images of the seabed, and which show the importance of phase and directionality in our perception of texture. A further extension is shown to give possible uses for image coding and object identification.

PRINCIPAL ABBREVIATIONS AND SYMBOLS.

λ	Fractal operator level.
D	Dimension.
ε	Operator iteration number.
u_ε	Upper level signal of operator for iteration ε.
l_ε	Lower level signal of operator for iteration ε.
V_ε	Volume produced by the operator at iteration ε.
β	Slope of log(power spectral density) .vs. log(frequency) graph.
Hz	Hertz, 1 cycle per second.
$N(\bar{x},\sigma^2)$	A Gaussian or Normal distribution with mean $\bar{x}$ and variance σ^2
$U(\bar{x},\sigma^2)$	A Uniform distribution with mean $\bar{x}$ and variance σ^2
PDF	Probability density function.
Λ	Wilks' lambda.
Σ	Summation symbol.
DC	Direct current, zero frequency component.
$E[.]$	Expectation operator.
D^2	Mahalanobis distance.
C	Covariance matrix.
log	logarithm, base 10.
ln	logarithm, base e.

CHAPTER 1

INTRODUCTION

1.1. Preliminary.

Texture is a characteristic of every surface. From this characteristic the human vision system gains much information with regard to an object or scene. It could be defined as a structure comprising a large number of similar elements or patterns without any single element being outstanding. For example the size of individual blades of grass in a field may be random, yet together they create a constant textural pattern. In trying to define texture, there is an immediate problem in that no universal description exists. This may be a reason why so many methods involving so many different techniques have been applied to measuring and describing it.

The textural properties of image regions provide important cues as to the orientation and depth of a surface. Changes in texture "coarseness" also provide important information about the angle and range. Directionality of the texture gives information on the angle of a surface and changes in coarseness indicate the possibility of an edge.

The study of texture is thus important in designing robotic vision systems where shape may be inferred from the 2D description [1.1]. In characterising a texture its description may be used for image coding [1.2]. In this manner the coded data is transmitted over a communication system allowing an effective reduction in bandwidth.

In studying texture for image analysis purposes there is also an important application in the understanding of the human vision system. In studying vision systems

Hubel and Weisel [1.3] showed that orientation played an important role in human and animal systems. Also some patterns were more easily detected than others. Julesz [1.4] investigated this aspect of the human vision system. He proposed the existence of a set of texture features or "textons" and adopted the view that texture segmentation is solely the result of differences in the first-order statistics of these local features. These textons are local features that give textural segmentation when textures have identical second order statistics. Julesz identified three classes of textons:colour, elongated blobs of given width, orientation and length, and the terminators of these elongated blobs. He found that textures could be detected preattentively (i.e. effortlessly and instantly) whenever a change in textons or texton density occurred.

Beck [1.5] presented results showing that texture segmentation occurs strongly on the basis of simple properties such as brightness, colour size and the slopes of contours and lines of the elemental descriptors of a texture or textural element.

In using models based on aspects of the human vision system Marcelja [1.6] argued that the representation of an image in the visual cortex involved both spatial and spatial frequency variables, and this partly involved the vision system performing a two dimensional Fourier Transform of the image. He proposed the use of a system due to Gabor [1.7] where a function is expanded in terms of symmetric (in-phase) and anti-symmetrical (quadrature) elementary signals which have the important property that they are maximally localised in space and spatial frequency. This led Pollen and Ronner [1.8] to suggest the use of pairs of Gabor filters covering various regions of the spatial frequency domain. These pairs of filters, giving both in-phase and quadrature terms, captured both spatial frequency and orientation properties [1.9]. The Gabor filter model has proved attractive as a mathematical attempt to model the human vision system, as the

90 degree phase difference between these filters also occurs for adjacent simple cells in the visual cortex.

1.2. Texture Segmentation and Discrimination.

This is the problem of splitting up an image into regions where each resulting region has a uniform texture. Studies have taken place on texture discrimination and texture analysis where measures have been sought to characterise a particular texture in the absence of others. Texture segmentation usually means separating textures in the presence of others and the two problems are not the same.

There are two approaches to segmentation of textures, namely those based on statistical measures and those based on structural measures. One of the more common statistical methods is that due to Haralick et al [1.10]. In this method estimates are made of the second order joint conditional probability density functions $f(i,j \mid d,\theta)$. Each one of these functions represents the probability of going from grey level i to grey level j given the intersample spacing, d, and the direction θ. The estimated values are written in matrix form, the co-occurrence matrix. From a co-occurrence matrix 14 texture features are extracted. Since this effectively computes a matrix for every value of d and every value of θ, storage and extraction of relevant features is a problem. Thus usually small values of the displacement (d) are used [1.11], [1.12] and if opposite directions are assumed the same then only the upper half of the co-occurrence matrix needs to be stored as it is then symmetric.

Not all the 14 features are usually used, Conners and Harlow [1.13] used five and obtained good segmentation on a range of Markov generated textures. Zucker and Terzopoulos [1.14] put forward statistical measures for extracting the spatial relationships

that best capture the features of the textures when using the co-occurrence method. These are essentially measures of the association between pixel grey levels, and are based on a chi squared statistic. Selkainaho et al [1.15] put forward another statistic, the kappa statistic, which was claimed to give a better indication of periodic structure.

A generalised co-occurrence matrix (GCM) [1.16] can be formed where features other than the grey level value of the image are used in forming the matrices.

Peitikainen and Rosenfeld [1.17] used the GCM with features derived from edges within the image. They measured curvature, average grey level, distance, contrast difference and slope difference between a pair of facing edges. The texture measures were then derived from the first order probability densities of these properties. Edge detection plays an important role in the perception of texture [1.18].

Khotanzad and Chen [1.19] used features derived from simultaneous autoregressive random (SAR) fields. They measured three SAR parameters in the horizontal and vertical directions and three in the diagonal directions. To these features a Sobel edge detector was applied, the resulting features were then smoothed using varying size windows. From these measurements an unsupervised system was used to produce good segmentation results on a variety of natural textures. They compared their results to six features derived from the grey level co-occurrence method (GLCM) and found that in some instances this latter method completely missed some of the texture boundaries.

Faugeras and Pratt [1.20] showed that use of Laplacian and Sobel edge detectors acted to decorrelate texture fields and produced separable measures between several groups of two texture problems. The separation measures used were based on the Bhattacharyya distance measure which gives a measure of the separation between two Gaussian distributions.

In the frequency domain several approaches to texture characterisation and segmentation have been used. Laws [1.21] proposed a set of masks which are convolved with the image under examination. These masks produce outputs which are summed over a region producing a "texture energy measure", and have been used by others in various forms. For example, Harwood et al [1.22] used Laws masks and formed ranked masks from them. He then used a rank correlation procedure to measure the differences between texture regions.

D'Astous and Jernigan [1.23] used several measures from the frequency domain as texture segmentation features. For example, measures such as the relative strength of the peak frequency (excluding DC), curvature of the peak frequency computed by the value of the Laplacian at the peak frequency, distance of the peak frequency from DC, and angle of the peak frequency. Further measures such as isotropy and circularity of the textures were computed. They concluded that these features performed better than traditional summed energy measures and co-occurrence measures. The comparisons were made on single textures and not on multi-textured images. An obvious problem when dealing with multi-textured images is that they would need to be segmented first, before being able to perform a possible classification.

Ikonomopoulos and Unser [1.24] used a technique based on directional filters to classify a set of separate textures. They effectively showed that directionality can act as a suitably powerful feature to allow classification of a set of textures; a point which plays an important role in the work of this thesis.

The design of a set of universal filters to achieve segmentation of a wide range of textures has been sought in recent years. In this the work of Knutsson and Granlund [1.25] is important in that their designs were based on models of the human vision system.

Furthermore these general kernels have been transferred onto a fast image processor system [1.26] and have achieved good segmentation results. They developed a set of in-phase and quadrature filters enabling estimates of the local frequency and orientation to be made. The outputs of the set of quadrature filter pairs are combined to produce a response at a given point in the image. In this way the outputs resemble the response of the cortical cells in giving a measure of frequency and orientation at a point. The use of quadrature filters also allows for an estimate of the phase at a point in an image. The importance of the phase in an image depends on the image [1.27]. Eklundh [1.28] showed that phase did not appear to yield any useful information as a texture measure, although it is important when describing images with strong line and edge content.

In further developing the quadrature filter idea and also basing models on the human visual system, Gabor filters have been studied extensively. Daugman [1.29] showed that these filters achieve the maximum joint resolution in the conjoint 2D visual space and the 2D Fourier domains. Furthermore the Gabor functions have profiles which closely match those of the great majority of the mammalian cortical simple cells [1.30]. The Gabor filters are Gaussian modulated sinusoidal gratings in the space domain and shifted Gaussians in the frequency domain. Clark et al [1.31] have shown their use in texture segmentation based on frequency and orientation. Further, in another paper [1.32], they show the detection of phase discontinuities between two regions of the same texture . This is important since although the eye sees the textures as the same it readily spots the phase discontinuities. The majority of all texture segmentation systems would not detect such a phase change, however.

Segmentation algorithms based on pyramid structures and multi-resolution have been developed. The pyramid processing idea was developed by Burt and Adelson [1.33].

In this approach the data consists of a set of images comprising different spatial resolution versions of the original image. In this segmentation system, class membership is passed from a lower to a higher spatial resolution. In moving from a higher to a lower resolution, spatial averaging takes place. The spatial averaging may be simple averaging or more complex kernels may be used. Different pyramid types exist. The Gaussian pyramid consisting of multi-resolution low pass filtered images, and the Laplacian pyramid consisting of multi-resolution bandpass filtered images.

Sher and Rosenfeld [1.34] have used the pyramid idea to segment a wide range of textures taken from Brodatz [1.35]. In their method the computation proceeds as follows:

a) A grey level image is input to the bottom level of the pyramid, one pixel per cell.

b) The intensity of eac cell is computed by pointwise multiplying the grey levels of its 16 children on the level immediately below it with a four by four kernel.

c) Each cell then decides whether it is a "spot" by comparing its own intensity with the intensities in its three by three neighbourhood. A spot is a cell whose intensity is either higher or lower than all the intensities of its eight neighbours.

d) Every spot computes the contrast between its intensity and the average of its neighbours' intensities. The contrasts of parents and children are compared to select "best" spots.

e) Each such spot serves as the "root" of a tree growing process which links up children with their parents according to their intensity similarity and spatial proximity.

f) The result of this linking process at the bottom level of the pyramid is the set of pixels assigned to the spot. This set defines the compact region associated with the spot.

Other authors' have used the multi-resolution approach. Instead of using the

original grey level values in the image other feature images may be used as the starting point. Spann and Wilson [1.36] used a quad-tree approach and achieved excellent boundary accuracy on Gaussian images using an unsupervised system. Unser and Eden [1.37] used a similar approach, again unsupervised. In this work local texture properties were extracted using local linear transforms that have been optimised for maximal texture discrimination. Local statistics (texture energy measures) were estimated at the output of an equivalent filter bank by means of a non-linear transformation (absolute value) followed by an iterative Gaussian smoothing algorithm. This procedure generated a multi-resolution sequence of feature planes with a half octave scale progression.

A new approach to texture representation and discrimination is presented by Chen and Wong [1.38]. A texture is represented by a set of frequency diagrams for various texture features at different resolution levels. Depending on the nature of the features, two kinds of diagrams can be put forward. Line frequency diagrams represent features as average grey levels and magnitude of the gradient in the neighbourhood. Circular plots represent directionality of the gradient. By keeping these diagrams at different resolution levels, as much as possible of the texture information, both structural and perceptual aspects, can be preserved. A metric, defined on the unfolded diagrams, is able to differentiate or associate two textures where human perception fails. For example, structurally different looking textures are perceived similar by the metric at high resolution and as resolution level decreases, their difference becomes apparent. The authors' used this metric as a nearest neighbour rule for multi-class texture classification [1.39].

1.3. Fractals in Image Analysis and Synthesis.

Consistent with the multi-resolution approach is the idea of fractals. This plays a

strong role in the present thesis. The term fractal was coined by Mandelbrot [1.40] and he showed that fractals had uses in many areas covering many disciplines. In essence he argued that it was not best to use Euclidean geometry to model nature, "*...clouds are not spheres, mountains are not cones, coastlines are not circles, and bark is not smooth...*". By identifying a family of shapes called fractals he also identified with them a fractal dimension. In Euclidean geometry a straight line lies in 1D and has a dimension of 1, as it breaks up and becomes more wrinkled (or noisy) its fractal dimension increases until eventually it is so irregular that it fills the whole 2D plane. When it is a solid square it has a dimension of 2, but it has evolved from the integer value 1 through real values to 2. Similarly it may then evolve through an infinity of real values from 2.0, through rolling countryside to jagged mountains until it arrives at a cube of dimension 3.0. In this way surfaces (and thus textures) may be viewed and modelled.

The idea behind fractals is that as finer detail is examined in a fractal signal or surface it then effectively repeats itself. This repetition with scale may be exact giving rise to the term self-similarity. An excellent (and popular) example of this is found by examining the "Mandelbrot" and "Julia sets". These are obtained by iterating the function $x_{n+1} = x^2_n + c$ where x and c are complex. Some spectacular images are produced when the real and imaginary parts of the iteration are plotted as colours in the Cartesian plane, the real taking one axis and the imaginary the other. On zooming in on the set (that is looking at a region in more detail), more stunning detail emerges, eventually arriving at the stage where the original returns. A modification to self similarity is that of statistical self similarity. This is the form most often found in nature. In this, the signal repeats itself with scale, but not exactly, only in the sense that zooming in on the signal shows that it has reproduced itself in a form where the statistical and thus spectral properties are similar

from one scaling to another. In practice of course, the amount of zooming is finite and the idea of a band limited fractal emerges.

Many fine examples of fractal images have appeared in the literature, [1.41] and some in the cinema as in "Star Trek II: The Wrath of Khan" and "The Last Starfighter". The idea of using the fractal model is appealing since it has appeared useful in modelling natural phenomena [1.42]. The question arises in respect of the work of this thesis as to whether the basic fractal model can be extended to provide information from which sidescan sonar data can be synthesised and eventually analysed.

In applying fractal ideas to image analysis and texture, the literature is relatively sparse. Some of the first examples appeared in 1984. Pentland [1.43] showed the segmentation of several images - an aerial view of San Francisco, a picture of a mountain range and a picture with a mug and a chair in it, so he did not restrict himself to texture. He broke the images up into 8 by 8 pixel blocks and computed the fractal dimension of each block by estimating the power spectrum within the block. The power spectrum is proportional to f^{2H-1} where f is frequency and H is related to the fractal dimension by $D = 3-H$ [1.41]. He thus estimated the fractal dimension using a regression on the log plot of power against frequency. He then constructed a fractal histogram which showed the number of occurrences of a particular dimension. Using the valleys in the histogram to act as threshold points with which to segment the original image, he obtained good results that could not be obtained by thresholding image intensity. He further showed that when the images were averaged to produce smaller images and the fractal estimation process was repeated, similar segmentation results were obtained showing the stability of the process over a change of scales.

At about the same time, Peleg et.al. [1.44] showed a scheme (which is further

described in chapter 2) where the area of a grey level surface is measured at several resolutions. This area decreases at coarser resolutions since fine details that contribute to the area disappear. Fractal properties of the image were then computed from the rate of this decrease in area and were used for texture comparison and classification.

Nakayama et.al. [1.45] used estimates of the fractal dimension to segment meteorological satellite image data. However the images were broken into 64 by 64 blocks and the results, although successful as far as they went, really required further processing. In contrast Medioni and Yasumoto [1.46] found that the fractal histogram model was not very successful at segmenting textures. They argued that placement rules would also be required and that then it would also be possible to synthesise and therefore classify and segment textures.

In 1987, Keller et.al. [1.47] showed how the fractal dimension could be used to compute scene characteristics based on the outlines of dominant silhouettes in the image. They further confirmed the stability of the fractal dimension under a variety of conditions and developed a new constant, the Holder constant, related to the fractal dimension which helped in the determination of distances in natural scenes. In 1989 [1.48] they showed the segmentation of some natural textures from Brodatz album. They used block sizes of 32 * 32 and used the concept of lacunarity in their measurements. The term lacunarity was first used by Mandelbrot and derives from the Latin "lacuna" meaning gap. He used this to describe the size of gaps or holes in textures (specifically in describing images of galaxies). This appears to be similar to the use of phase in helping to describe the positions of primitives in an image. Many images can have the same fractal dimension but differ in lacunarity. Using the concepts of lacunarity they found they got improved results for segmentations which had previously used only fractal dimension.

In image synthesis fractals have produced many fine examples of computer generated imagery. However, little work has been done on using fractals for texture generation. Dodd [1.49] synthesised some natural textures in both monochrome and colour. He used principal component space to decorrelate the RGB (red-green-blue) colour space and then modelled the fractal dimensions of these principal components. When these components were transformed back to RGB space a realistic image was produced. Much of the work on image synthesis has concentrated on rendering surfaces once the model has produced an image. An important paper in the modelling work was that of Fournier et.al. [1.50] who showed that a stochastic method could be used to generate realistic images. This stochastic method was shown to give good approximations to fractal models and the authors' used it to generate images for some films.

Dennis and Dessipris [1.51] suggested the use of anti-aliasing filters to overcome the problem of errors in estimating the fractal dimensions of textured images, anti-aliasing being used to restrict the measurement to the range of scales over which a texture was likely to be fractal.

At this point the work of Gagalowicz [1.52] should be mentioned. He has produced some excellent examples of synthesised textures based on measurements from the co-occurrence matrices of real textures. There is here a close relationship between some of the measurements derived from co-occurrence matrices and the fractal dimension. This is explored in chapter 3.

1.4. Statistical Aspects.

In regard to the statistical aspects of texture segmentation, some papers have explored further the problems of feature selection. For example, D.C. He et.al. [1.53] used

the linear discriminant and a procedure based on the Mahalanobis distance to select an optimum set of features from a set of 173 features. The algorithm allowed features previously rejected to be put back into the scheme for re-evaluation. From the set of 173 features they selected 10 and obtained classification accuracies of the order of 95% when classifying isolated textures.

Liu and Jernigan [1.54] used 28 features from the frequency domain and then Wilks' criterion (chapter 4) to find an optimal subset. In their analysis they found that using all 28 features did not give the best results; the optimum number of features they found was 7, achieving a 97% successful classification rate, again when applying classification to isolated textures.

Recently, Siedlecki and Sklansky [1.55] introduced a genetic algorithm for large feature sets (greater than 20). They claimed that this gave better results for finding the optimum subsets than the branch and bound method [1.56].

Hsiao and Sawchuk [1.57] used supervised methods to segment textures, with the emphasis on boundary detection. They also report results of an unsupervised scheme [1.58]. In these methods, after initial processing, a probabilistic relaxation scheme [1.59] is implemented to help find the texture edges more accurately; in effect making efficient use of context within the image.

1.5. *Thesis Layout.*

This thesis presents a study of texture and texture segmentation based on fractal ideas. The features used are presented to both supervised and unsupervised recognition schemes for analysis and processing of the textures.

Chapter 2 presents an operator for obtaining texture features. The operator has its

foundations in fractal measurements where it was used to determine the fractal dimensions of lines and surfaces. Its use in 1D and 2D is shown and its effects on signals in both the frequency and statistical domains is presented.

In chapter 3 the use of fractals in the synthesis of textures pertinent to studies of sea bed textures is presented. Several models are suggested and examples given. The importance of phase in images is shown and also how it may be measured. An example of the use of fractals for interpolation and extrapolation of images is shown.

Supervised pattern recognition using statistical methods is presented in chapter 4. This gives a background to the discriminant model. The questions of the non-Gaussian nature of the data, training area size, feature selection and use of the linear and quadratic discriminants are addressed. The use of the discriminants in classifying multi-textured images is shown with the features found using the operator of chapter 2.

The subject of chapter 5 is unsupervised pattern recognition. In this chapter the use of uncorrelated features obtained by principal component analysis of the original feature set is shown. A scheme is developed using some of the techniques from chapter 4 to effect an unsupervised segmentation based on an iterative quadratic discriminant.

Finally, chapter 6 presents conclusions and suggestions for further work.

1.6. Authors' contribution.

In this thesis it is believed that several new topics are presented for the analysis and synthesis of multi-texture images. The main contribution is the use of the fractal operator combined with the subsequent statistical processing to produce segmentation boundaries in multi-texture images. The operator itself is not new, however it is felt that considerably more efficient and novel use of it is made in this study. Previous authors'

have used such an operator to determine the fractal dimension of a surface or image region. This thesis highlights the fact that such an approach is too limited when dealing with natural texture images. When the operator is used in several directions then the feature images produced provide powerful texture discriminators when input into both supervised and unsupervised processing schemes.

In the supervised statistical processing system, it is shown that the Gaussian distribution is a good model even when the feature data is non-Gaussian. The performance of linear and quadratic discriminants is examined and while there is little to choose between them in terms of classification accuracy, it is shown that the quadratic discriminant is more robust for non-Gaussian data. The selection of training areas is examined and a new adaptive processing algorithm is presented. This allows the effective training areas to grow as the analysis proceeds, producing a more accurate classification with little increase in processing time. The effectiveness of two feature selection algorithms is reviewed using the fractal features as input data.

In the unsupervised processing system, it is shown that the transformation of the original directional fractal features into principal component space provides more efficient features than the original set. A processing system is presented which uses the largest principal component features to identify automatically training areas and subsequently use the adaptive classification system to produce results with low classification errors. Most of the examples presented are with reference to sidescan sonar images of the seabed, which must be processed automatically due to the large amounts of data acquired, the segmentation boundaries being required for navigational purposes and geological and oceanic studies.

In studying texture, some new models for texture synthesis are presented based on

fractal ideas. It is shown that some types of texture (including those produced from sonar records) can be modelled accurately using the fractal concept. The effects of anisotropy, shading and the production of multi-texture images are considered. All of these models use a random phase component. However, it is shown that some textures require a detailed knowledge of the phase and cannot be reconstructed using a random phase. Extensions to the fractal model are given. Both frequency interpolation and extrapolation are explored and shown to have potential use in texture reconstruction and texture magnification.

CHAPTER 2

A MEASURE OF TEXTURE BASED ON THE FRACTAL DIMENSION

2.1 Introduction.

As was discussed in chapter 1, texture and its description is complex and no single universal definition exists. Texture in common usage refers to the character of a woven fabric or textile. It is also used to describe any arrangement of particles or constituent parts of any material such as wood, concrete or metal. Texture is often correlated with tactile senses and described as smooth coarse or grainy [2.3]. It is this last aspect, that of surface roughness, which lies at the basis of fractal dimension and will be considered further in this chapter.

The problem of texture image segmentation may be viewed as that of transformation of an image into one or more other images, so that these resulting feature images may be separated by statistical or spectral methods. The need for this transformation arises since the original image rarely has statistics or spectra that allow an immediate segmentation. Segmentation in this chapter implies separating areas of texture that the eye sees as different. For example, in extreme cases the texture may be the same throughout the image but it may be necessary to delineate areas where the directionality differs and certainly to delineate areas where the textures themselves differ. Furthermore the delineation should not be affected by a texture showing a change in grey level over

different areas due to perhaps lighting changes or the change in response of the recording instrument.

The reason for regarding directionality as important is that much of the application of this work is in mapping areas of the seabed and directionality is used in monitoring changes with time.

It was decided to consider some fractal ideas for texture description in exploring texture segmentation algorithms. This decision was made because the use of fractals in producing realistic images of natural scenes is impressive. However, their use in analysing images is more limited. Relatively few authors have explored the use of fractals in texture analysis. Keller et al [1.48] showed its use in segmenting natural scenes, Pentland [1.43] measured the power spectrum over small image areas and used the fractal dimension as an image descriptor. Medioni [1.46] found that the fractal dimension was not useful as a texture descriptor and Peleg et al [1.44] used the change in area with resolution as a measure of fractal dimension to classify textures.

Several techniques have been formulated for measuring the fractal dimension of a texture, namely

a) Perimeter - Area measurements,

b) Box measurements.

c) Change in statistic with resolution.

These are all two dimensional(2D) measurements and none takes directionality into account. It was decided that a technique should be used that allowed directionality to be a feature in the measurement, since it is an essential element of texture description. To this end it was decided to use a set of one dimensional(1D) measurements, each measurement differing in direction. One procedure, that of the "sausage method" [1.40]

looked useful. In this technique the change in length of a curve is measured at differing resolutions. The technique is often used by mathematicians to measure the length of a non-deterministic curve. Mandelbrot [1.40] has shown that for a fractal line, the sausage method could give a measure of its fractal dimension. Peleg [1.44] showed that the technique could be used in the 2D case of textures. However, several problems were apparent. While the textures all had differing fractal dimensions they were each measured in isolation. This is an important fact for as Professor Granlund [1.25] has pointed out "*.... verifying a texture difference when the borders are known is a much simpler task and also one of little practical use....*". Also no account had been taken of directionality. This is important since a texture may be considered as an anisotropic fractal. This idea is explored further in chapter 3. Further, the fractal dimension is measured over an area (box) and is thus an average. Average here means that the texture may be made up of more than one fractal and in more than one direction. Thus the resulting segmentation would be in blocks and not suitable for the present study. The computation required is also extensive and requires of the order of fifty iterations. The following topics were therefore addressed.

a) Does the fractal dimension have to be measured to provide a measure for segmentation ?

b) Could the number of iterations be significantly reduced ?

c) Could directionality be used as an additional feature measure ?

d) Could several textures be separated without knowing the boundaries of the textures ?

e) What resolution could be achieved in delineating texture discrimination ?

2.2 The quest for fewer iterations.

It was hoped that the addition of an extra "dimension" in the form of directionality would aid the discrimination of textures, and also reduce the need for examining the textures at so many resolutions. The multi-resolution approach is a powerful aid to discrimination [1.33] [1.36] but it is also extremely taxing on computing requirements, thus any decrease in the number of levels of resolution to be taken into account would be advantageous.

Further, since the idea of multi-resolution is at the heart of fractals and since it would be best to reduce the number of experiments on multi-resolution it may not be essential to determine the actual fractal dimension of the texture.

Briefly, the background to the measurement scheme is as follows. Mandelbrot [1.40] found that for many curves, their length could be approximated using the formula,

$$L(\lambda) = k\lambda^{1-D} \qquad (2.1)$$

Where $L(\lambda)$ is the length of the curve measured using a unit of length λ, k is a constant and D is the fractal dimension of the curve and is a constant for that particular curve. One of several methods he proposed for measuring this dimension was to consider points a distance λ from the curve (on either side) so forming a strip 2λ wide. The area of this strip divided by 2λ giving an approximation to the length. As λ decreased, the length increased. Extending this to a surface in two dimensions, i.e. an image, produces an upper level at distance λ above the surface, and a lower level at distance λ below the surface. Computationally, given an image $g(i,j)$, where i, j represents a pixel in the image, then the initial values of the upper and lower images (blankets) are defined as

$$u_0(i,j) = l_0(i,j) = g(i,j)$$

where u and l represent the upper and lower blankets. For any point (i,j), the blankets are computed as

$$u_\varepsilon = \max([u_{\varepsilon-1}(i,j)+\lambda], [u_{\varepsilon-1}(m,n)]) \qquad (2.2)$$

$$l_\varepsilon = \min([l_{\varepsilon-1}(i,j)-\lambda], [l_{\varepsilon-1}(m,n)]) \qquad (2.3)$$

where (m,n) is the position of a pixel surrounding (i,j) in an eight connected neighbourhood. λ is the level by which the pixel at (i,j) is raised, and ε is the iteration number.

The volume enclosed between these two layers is

$$V_\varepsilon = \sum (u_\varepsilon(i,j) - l_\varepsilon(i,j))$$

The change in volume between iterations ε and ε-1 is

$$\delta V = \frac{V_\varepsilon - V_{\varepsilon-1}}{2\lambda}$$

where the divisor, 2, accounts for the upper and lower layers. Mandelbrot has related the area to the fractal dimension in a similar expression to that for the length, namely

$$A(\lambda) = k\lambda^{2-D}$$

where the exponent has merely increased by 1, ie the dimension has increased by 1.

To illustrate the technique, consider two of the 1D signals from the set of fractal signals shown in figure 2.1. These two signals represent fractal lines of dimension 1.2 and 1.5. They were constructed by the techniques discussed in chapter 3. Applying the above ideas to these signals produces the results shown graphically in figure 2.2, where *log(δA)* has been plotted against *log(iteration(ε))*. δA being the change in area between iterations. The slope of these lines is (1-*D*) and when measured they give fractal dimensions(*D*) of 1.22 and 1.51 respectively, a satisfactory approximation. If the value of the level(λ, currently set to 1) is well within the range of the amplitudes of the signal then many different values could be used for λ. If the signal is a true fractal then the result should be independent of the starting value of λ, since true fractals are scale independent.

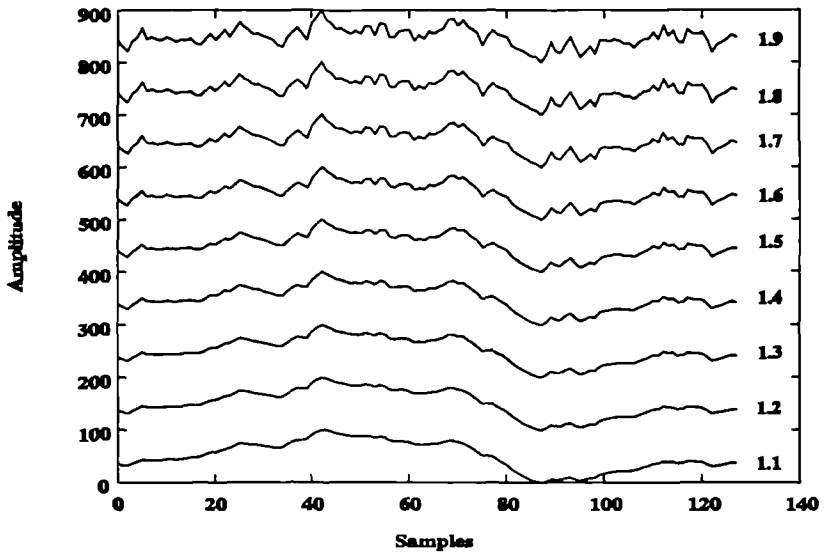


Figure 2.1. Fractal Signals of dimensions 1.1 to 1.9 in 0.1 steps.

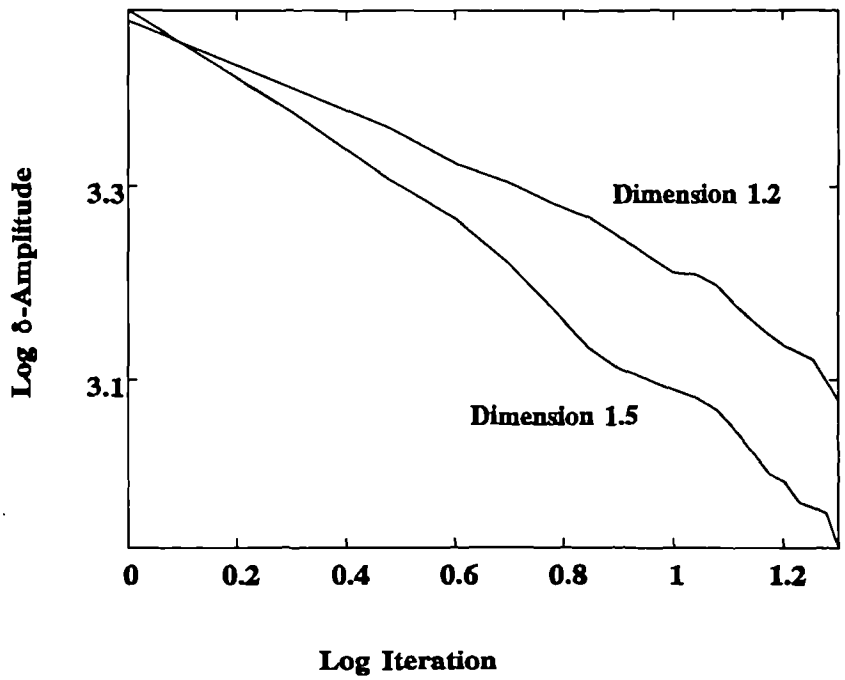


Figure 2.2. Log(δA) .v. Log(ε) for signals of fractal dimension 1.2 and 1.5.

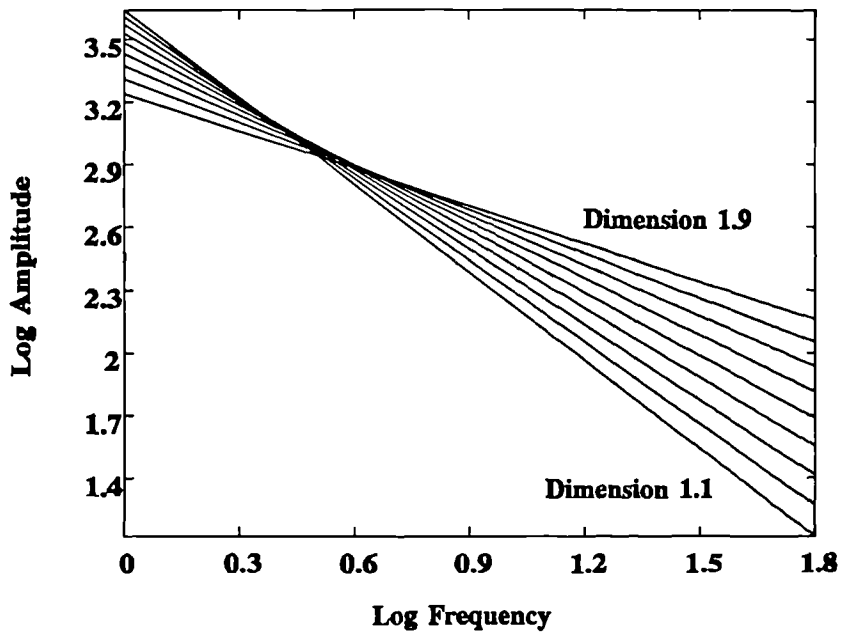


Figure 2.3. Log(Amplitude) .v. Log(frequency) for signals of Figure 2.1.

The signals of figure 2.1 have quite different spectra as shown in figure 2.3. In measuring the fractal dimension, the slope of the *log(δA)* .v. *log*(ε) line is (1-*D*), and *D* is related to the power spectral density [1.41] by

$$D = \frac{5 - \beta}{2}$$

where β = the slope of the line produced by plotting the *log(power spectral density)* against *log(frequency).* The technique thus approximates the estimation of the power spectral density by successive iterations on the signal. To investigate the procedure further, consider the set of 1D fractal signals shown in figure 2.1. These have fractal dimensions from 1.1 to 1.9 in steps of 0.1, log-log plots of their amplitude spectra against frequency are shown in figure 2.3. These fractal signals represent a broad range of signals covering a wide domain of spectral amplitudes and frequencies.

The action of this min-max (or fractal) operator (hereafter referred to as the operator) in both the spectral and statistical domains was investigated. This was to determine whether a reduction or even elimination of the number of iterations is possible, since these are only required to provide an estimate of the fractal dimension.

2.2.1. The operator in the frequency domain.

The initial input signal is operated on with the max and min operators (equations (2.2) and (2.3)) to produce the upper and lower signals and the difference between these produces the transformed signal. Another application of the max and min operators to these upper and lower signals then produces another set of upper and lower signals from which the next difference signal is produced, and so on. This operation is shown schematically in figure 2.4.

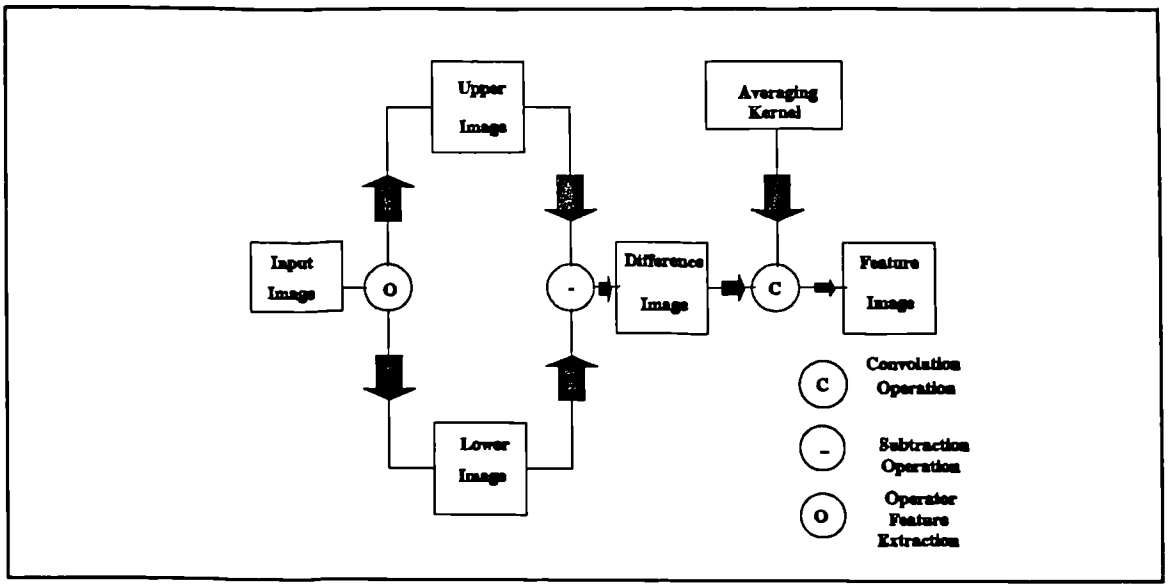


Figure 2.4. Schematic representation of feature image production.

In order to visualise the effect in the frequency domain, a fractal signal of dimension 1.2 was selected from the set shown in figure 2.1. Five iterations of obtaining the difference signal were performed and the spectra of the original signal and these iterates were examined. Figure 2.5 shows the log-log plots of amplitude against frequency for all six signals.

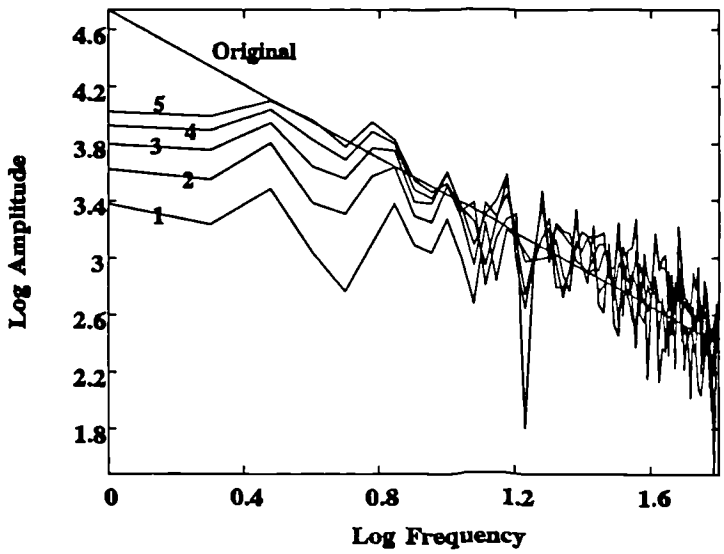


Figure 2.5. Log(Amplitude) .v. Log(frequency) plot, original and 5 iterates.

As can be seen from figure 2.5, the first iteration has removed a considerable amount of the low frequency part of the signal, in effect acting as a high pass filter. The successive iterations gradually restore the low frequencies but the changes in spectral content after the first iteration are relatively small. To illustrate this change, the results in table 2.1 show the power content (relative to the first iterate) of each of the signals. More significant is the power change as the iterations progress. After the first iteration, the power changes between the difference signals produced by the operator vary according to the spectral content of the original signal. The greater the high frequency content the less the change in power between iterates. Further the change in power is almost solely due to the low frequency and DC components. Figure 2.6 shows the results for five iterations on all the fractal signals depicted in figure 2.1. This illustrates the change in power (relative to the first iterate) with the number of iterations.

Iteration	**1.1**	**1.2**	**1.3**	**1.4**	**1.5**	**1.6**	**1.7**	**1.8**	**1.9**
1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
2	3.1	2.7	2.5	2.3	2.0	1.8	1.7	1.5	1.4
3	5.8	5.0	4.3	3.7	3.1	2.6	2.2	1.9	1.7
4	8.9	7.6	6.4	5.3	4.2	3.3	2.7	2.2	1.9
5	12.2	10.3	8.4	6.8	5.2	3.9	3.0	2.3	1.9

Table 2.1. Relative power levels for fractal signals 1.1 .. 1.9, first 5 iterates.

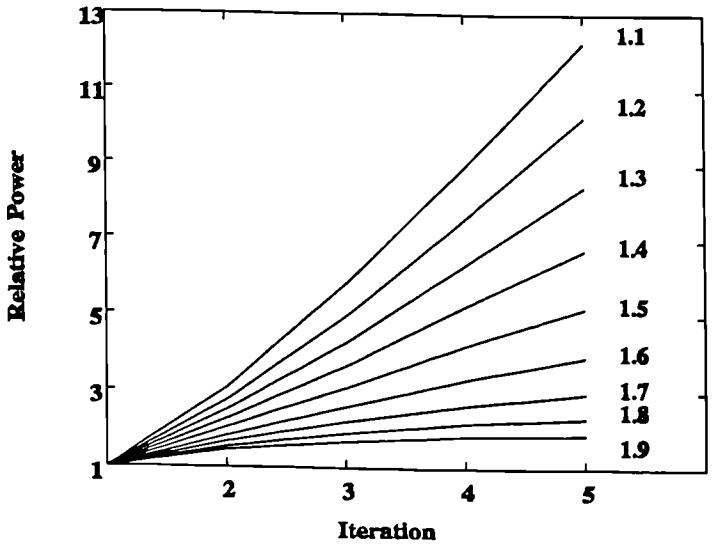


Figure 2.6. Relative power levels for first 5 iterates of signals of Figure 2.1.

The evidence for reducing the number of iterations is further illustrated by the use of the chirp signal shown in figure 2.7. Here the original signal has frequencies from 1 to 32 Hertz, gradually moving from low to high frequency over all 512 samples, the amplitude being kept constant. By performing successive iterations on this, the results shown in figure 2.8 are obtained. These show the effect in the time domain, where the gradual amplification of the low frequencies and increasing difficulty of tracking the high frequencies is illustrated.

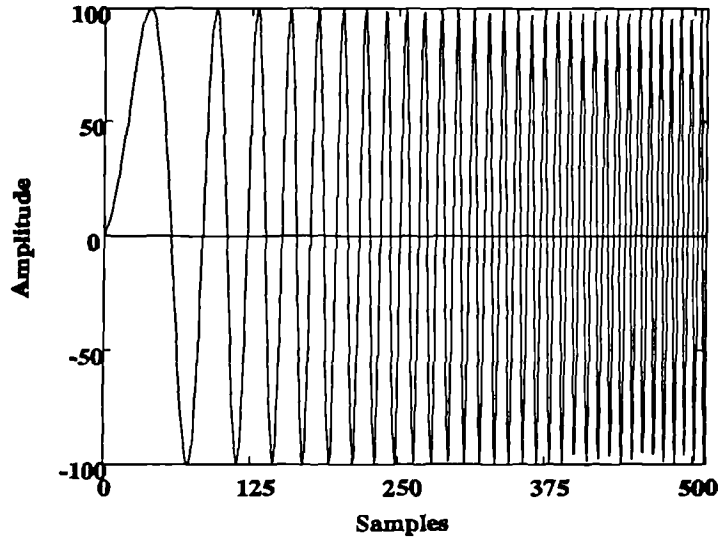


Figure 2.7. Chirp signal, frequencies 1..32 Hz.

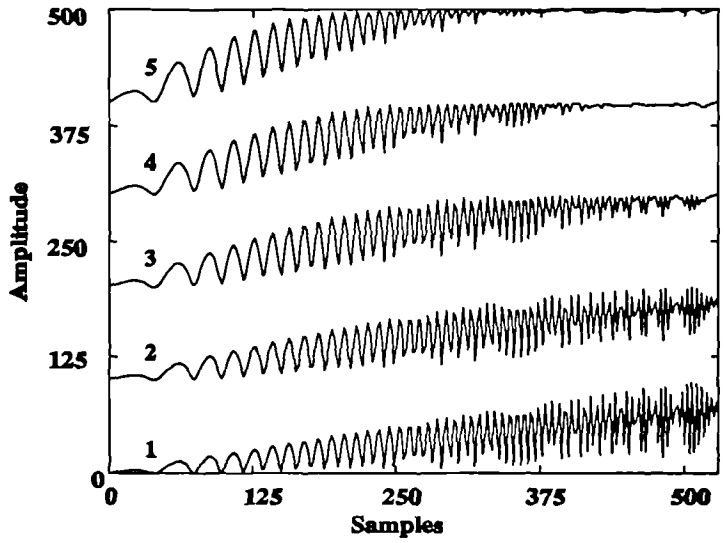


Figure 2.8. Time domain plot for first 5 iterates on signal of Figure 2.7.

(a) The effect of the level(λ).

All previous results have used a value of 1 for the operator level,λ, however the effects of this apparently arbitrary level are now considered. For a true fractal the operator level(λ) should not affect the results when measuring fractal dimension since it is invariant to scale. In practice the limitations of bandwidth and quantisation must be taken into account. In effect the signals are considered as band limited fractals. Furthermore, since the idea of measuring fractal dimension for texture classification is questioned then so is the need for a large number of iterations. The use of many fewer iterations will be considered, using the signal produced by the iteration as a new 'feature' signal. The effect of the value of λ must now be considered. To study the effect of λ, the factor λ in equations (2.2) and (2.3) is now considered for some values other than 1. If the experiments of the previous section are repeated, the effect of varying λ can be assessed. In these examples, λ is given a value greater than 1 and the difference signals found for a set of iterations. Figures 2.9a,b,c,d show the results of successive iterations on the chirp signal of figure 2.7. The values of λ were chosen as 3, 7, 11, and 15 respectively. For each set of iterations λ was of course kept constant at one of these values. This figure again illustrates that after the first iteration the tendency is to amplify the lower frequencies with increased inability to cope with the high frequencies. Furthermore it also shows that as λ is increased more lower frequencies are ignored.

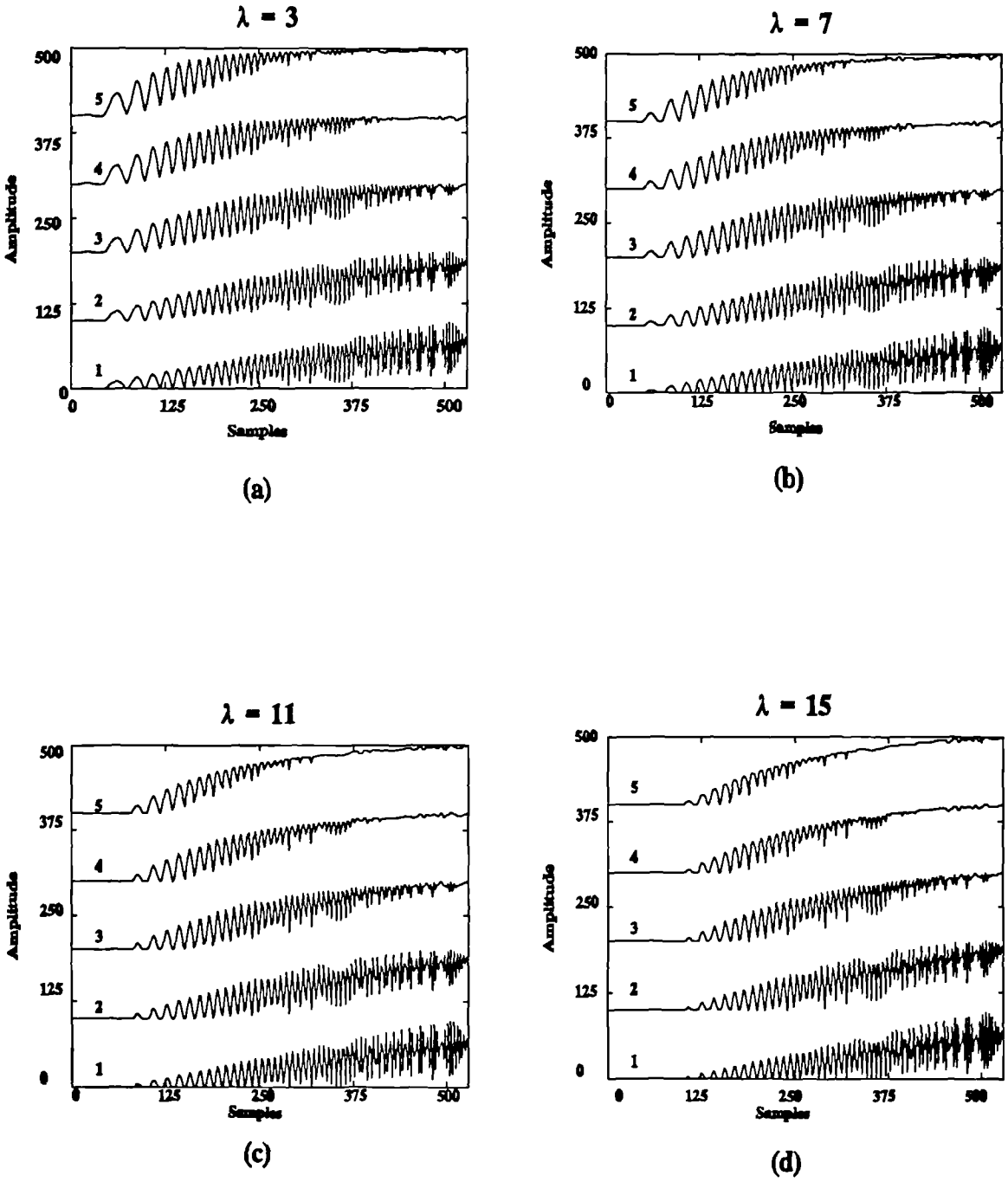


Figure 2.9. Successive iterates using varying λ for chirp signal.

It appears at this stage that the operator is acting in a similar manner to a differentiator and it is therefore worth considering more of its fundamental properties in the light of this.

Consider a sine wave which has one cycle covering 512 samples and has an amplitude of 100 units. One iteration of the operator (with a λ value of 0) produces the result shown in figure 2.10. The result is similar to a full wave rectified cosine signal.

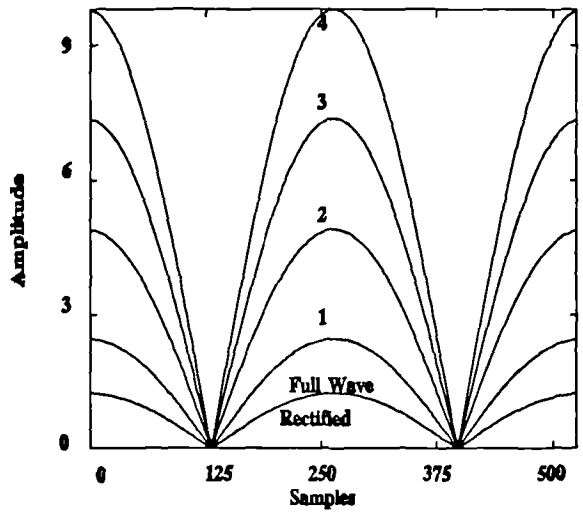


Figure 2.10. Theoretical signal and first 4 iterates, $\lambda = 0$.

The original signal may be represented by :

$$y(t) = 100 \sin (2\pi/512)t$$

Differentiating this gives :

$$y'(t) = z(t) = 1.227\cos(\pi/256)t$$

The full wave rectification of this produces a signal which may be represented as

$$z(t) = 1.227\left[\frac{2}{\pi} - \frac{4}{\pi}\sum_{k=1}^{\infty}\frac{(-1)^k}{4k^2-1}\cos\frac{2\pi kt}{256}\right] \qquad (2.4)$$

Which is obtained by considering the Fourier series expansion of the signal:

$$z(t) = 1.227\ \cos(\pi/256)t$$

This is a standard result and details may be found in many standard texts e.g. [2.1]. If the results of further iterations are obtained, then the relevant results shown in table 2.2 are produced.

Signal	Mean	Variance	Maximum	Minimum
Rectified	0.78	0.143	1.227	0.008
Iteration 1	1.56	0.571	2.454	0.008
Iteration 2	3.12	2.276	4.908	0.030
Iteration 3	4.67	5.096	7.361	0.068
iteration 4	6.21	9.011	9.814	0.120

Table 2.2. Parameters for theoretical signal and iterates, $\lambda = 0$.

This table shows the mean, variance, maximum and minimum for these iterates and the theoretical full wave rectified equivalent signal obtained by summing 50 terms of equation (2.4). From these results the following observations may be made.

The first iteration produces a result which is exactly twice the result from the theoretical signal (four times the result for the variance, since this is a second order term). This doubling arises because of the differencing of 2 signals, namely the upper and lower

blankets. As iterations continue, the values increase in direct proportion to the iteration number. This indicates that there is little point in using iterates higher than the first for this level of operator, since, although the mean is increasing, the variance is increasing proportionally and this would not give any benefits to a statistical classifier.

Consider now the results of a similar set of experiments where the operator level has the value 1. The results are shown in figure 2.11.

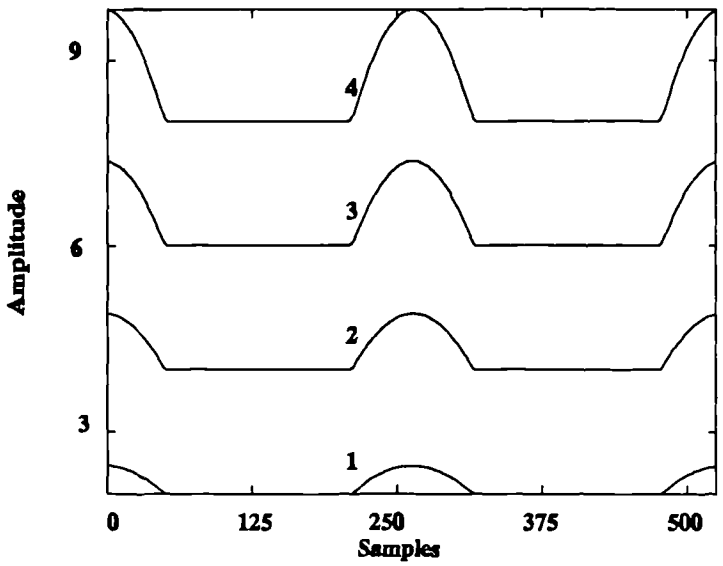


Figure 2.11. First four iterates, $\lambda = 1$.

This shows that the iterate is thresholded at a level of 2λ. Successive iterates increase the base level by a factor of 2λ at each iteration, as well as amplifying the signal. Table 2.3 shows the values of the mean and variance for this set of iterates.

Signal	Mean	Variance	Maximum	Minimum
Iteration 1	2.11	0.028	2.454	2.000
Iteration 2	4.23	0.114	4.908	4.000
Iteration 3	6.35	0.254	7.361	6.000
Iteration 4	8.46	0.446	9.814	8.000

Table 2.3. Parameters for first four iterates, $\lambda = 1$.

This time, although the variance is increasing it is not increasing as the square of the iteration, thus there appears to be a benefit in considering an iterate other than the first when λ is greater than 0.

Also from the above analysis, it should be noted that a maximum value of λ may be established. This maximum value occurs at *half* the maximum value of the signal for the first iterate, obtained with a λ value of 0, i.e. in the present case this would give λ_{max} = 1.227. At this value the resulting signal would be a straight line with a DC value of 2.454, the variance of course being zero. In effect, at this λ, this frequency of signal would be ignored. This is precisely the effect observed in the chirp signal where various low frequencies were ignored depending on the λ value used. This also explains the reduction in variance of increasing λ, due to the disappearance of the lower frequencies of the signal.

It may now be ascertained that a larger value of λ may be tolerated for higher frequencies in the signal. In fact the following relationship may be given:

$$\lambda_{max} = \max(A\omega)$$

Where A is amplitude and ω is the frequency. $A\omega$ is the product for any particular frequency component in the signal.

The effect of operator length is now considered. So far this has been fixed at 3. It is to be expected that if the operator length is increased a smoothing will take place and effectively give a bandpass characteristic to the operator, the operator itself ignoring the low frequencies, and the larger kernel size attenuating the higher frequencies. Figure 2.12 shows the results, again on the chirp signal, for a fixed λ value of 1 and operator kernel lengths of 3, 7, 11 and 15. The effect however is very similar to changing the value of λ i.e. increasing the amplification of the mid frequencies, ignoring the lower frequencies, and having difficulty in tracking the high frequencies. It is thus considered that a kernel size of 3 should be maintained.

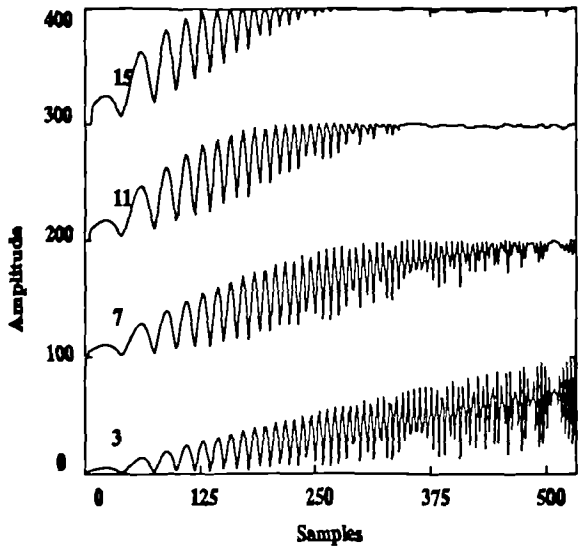


Figure 2.12. Effect of varying kernel size, decreasing from top(15) to bottom(3).

So far an experimental approach has been used to gain an insight into the effects of the operator on signals. A more formal approach to the frequency aspects of the operator is now presented.

2.2.2. Frequency analysis of the operator.

Since the fractal operator acts on three samples, let these three samples be represented by

$$\{ u_{t-1}, u_t, u_{t+1} \}$$

where the output sample is generated for time *t*.

Given an operator level of $\lambda = 0$, six possible cases exist for the generation of the sample output. For example, the maximum sample may be u_{t-1} and the minimum sample u_{t+1} giving rise to an output of $(u_{t-1} - u_{t+1})$. In the frequency domain this may be represented as,

$$\begin{aligned} u_{t-1} - u_{t+1} &= e^{i\omega(t-1)} - e^{i\omega(t+1)} \\ &= e^{i\omega t}(e^{-i\omega} - e^{i\omega}) \\ &= -2i\sin\omega . e^{i\omega t} \end{aligned}$$

This gives rise to an amplitude frequency response of $2\sin\omega$ on the sample at time *t*. All six cases are listed below, showing the responses for each case.

1)	$u_{t-1} - u_t$	$= e^{i\omega t}(e^{-i\omega} - 1)$	$= \cos\omega - i\sin\omega - 1$	::	$(2 - 2\cos\omega)^{1/2}$
2)	$u_{t-1} - u_{t+1}$	$= e^{i\omega t}(e^{-i\omega} - e^{i\omega})$	$= -2i\sin\omega$	::	$2\sin\omega$
3)	$u_t - u_{t-1}$	$= e^{i\omega t}(1 - e^{-i\omega})$	$= 1 - \cos\omega + i\sin\omega$	::	$(2 - 2\cos\omega)^{1/2}$
4)	$u_t - u_{t+1}$	$= e^{i\omega t}(1 - e^{i\omega})$	$= 1 - \cos\omega - i\sin\omega$	::	$(2 - 2\cos\omega)^{1/2}$
5)	$u_{t+1} - u_{t-1}$	$= e^{i\omega t}(e^{i\omega} - e^{-i\omega})$	$= 2i\sin\omega$	::	$2\sin\omega$
6)	$u_{t+1} - u_t$	$= e^{i\omega t}(e^{i\omega} - 1)$	$= \cos\omega + i\sin\omega - 1$	::	$(2 - 2\cos\omega)^{1/2}$

From this it is seen that cases 1,3,4, and 6 have the same amplitude response as each other, while cases 2 and 5 have the same response as each other.

Because of the fact that the operator uses the (maximum - minimum) values, this is always positive and equivalent to taking the absolute value, giving rise to only 2 phase responses, one for cases 1,3,4,6 and one for cases 2 and 5, instead of the six possible phase responses.

In order to verify the responses for these six cases, the response for each case was measured using a sine wave with frequencies in the range 0 to 128 cycles with 256 samples in each wave. Thus the frequency response up to the Nyquist limit was measured, $\omega=0..\pi$. In measuring these responses for each case, the operator was not used in its normal way. For example, for case 1, the output for sample t was generated from samples t-1 and t, irrespective of the maximum and minimum values. The responses for all six cases confirmed the theoretical results. The responses for cases 1 and 2 are shown below as examples, with the theoretical response overlaid. For cases 2 and 5 the theoretical results agreed exactly.

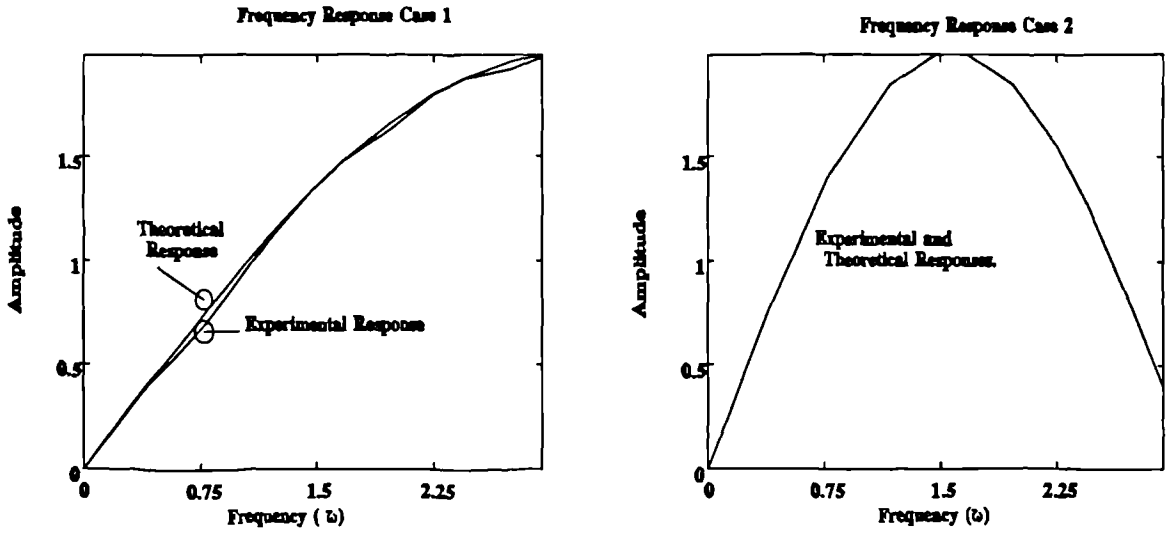


Figure 2.13. Experimental and theoretical frequency responses for cases 1, 2.

Thus given a signal it may be seen that the output at any sample point is dependent on the frequency of the signal at that point and also on the relative magnitude

of the signal at that point and the samples at either side of it (i.e. on the local derivative). Which case (of the six) is chosen by the operator at a point is determined by the local derivative, which in turn is governed by the operator level λ. To illustrate this consider the case for three sample values

$$u_{t-1} = 9.5,\ u_t = 9.0,\ u_{t+1} = 7.0.$$

For $\lambda = 0$, this would give rise to case 2, and frequency response $2\sin\omega$.

For $\lambda = 1$, this would give rise to case 4, and frequency response $(2-2\cos\omega)^{\frac{1}{2}}$.

In this way the input signal gives an output which depends on the input signal frequency at the point and on the local derivative at the point, which is governed by λ, the operator level.

To summarise these points, -

a) Given a signal, the output when the operator is applied is governed by 3 quantities.

i) The value of λ.

ii) The derivative at either side of the signal at time *t*.

iii) The frequency of the signal at time *t*.

b) If the value of the derivative at either side of the signal at time *t* is less than λ, the output will be zero (or more correctly 2λ).

c) If the value of both derivatives at either side of the sample at time *t* is greater than λ, then the output is from case 2 or 5, and the result is independent of λ.

d) If the value of the derivative from either one side or the other (but not both) of the sample at time *t* is less than λ, then the output is from case 1,3,4 or 6.

e) The value of λ greatly affects the resulting output, since this determines which filter is used. For two signals of the same frequency but different amplitudes (i.e.

different derivatives), then different values of output will be obtained depending on whether the chosen λ exceeds the slopes at different points. For example, consider the following situation, where the two signals are $A_1\sin\omega t$ and $A_2\sin\omega t$ ($A_1 < A_2$). The maxima and minima always occur at the same sample points for both signals when $\lambda = 0$ is used. However, if a value $\lambda > 0$ is used the output now depends on the slope at a sample point. For the smaller derivative signal ($A_1\sin\omega t$), when the slope is greater than λ the output is the same as for $\lambda = 0$, whereas when the slope falls below λ on only one side of the sample point t, the output changes to one from case 1,3,4, or 6. For the larger signal ($A_2\sin\omega t$), these changes occur less frequently since the derivative is greater overall by (A_2/A_1) times.

f) An operator has been presented which is sensitive to both frequency and derivative. Further, the output it produces can be controlled by careful selection of the value of the level λ of the operator. The output is not affected by changes in the mean of the signal, but will change when two similar signals of different amplification are compared.

This is useful for texture, since the same texture whose DC level is raised (due to illumination changes) will respond in the same way.

Having looked at the effects of the operator in the frequency domain, the effects in the statistical domain are now considered.

2.2.3 The operator in the statistical domain.

Since any features created by the operator with an image will ultimately be submitted to statistical methods for analysis, the statistical aspects of the operator and the effect of the iterative cycle on the statistics of the features produced should be considered.

In order for successive iterations to generate significant features, each feature or iteration must contribute new information to the feature vector. In the frequency domain there appears to be little change in the spectral content with successive iterations after the first. The effect of successive iterations is therefore investigated.

The first aspect to consider is the correlation of the features between successive iterations. If successive iterations are highly correlated then there will be little or no information content to be analysed by a statistical method and therefore little need to include several iterations. To investigate this aspect, consider the fractal signals shown previously in figure 2.1. Using a λ value of 1 and an operator length of 3, five iterations for producing the difference signal were performed. The correlations between successive difference signals were then computed, for example, iteration 1 with iteration 2, iteration 2 with iteration 3 etc.. The results are tabulated in table 2.4. From this it can be seen that there is a high correlation between successive iterations, the correlations increasing as the number of iterations increases and being large for all signals. The conclusion is that it would not be useful to use successive iterates as additional features for a statistical classifier. It may also be seen that as the fractal dimension and thus the high frequency content of the signal increases, the correlation for any particular iteration decreases.

	Fractal Dimension								
Iterates	**1.1**	**1.2**	**1.3**	**1.4**	**1.5**	**1.6**	**1.7**	**1.8**	**1.9**
1,2	0.844	0.808	0.787	0.779	0.762	0.740	0.723	0.710	0.702
2,3	0.923	0.897	0.877	0.864	0.848	0.830	0.811	0.800	0.794
3,4	0.962	0.951	0.940	0.927	0.912	0.891	0.874	0.869	0.862
4,5	0.972	0.964	0.955	0.944	0.929	0.912	0.904	0.902	0.902

Table 2.4 Correlation between iterates for fractal signals 1.1 to 1.9.

The statistical aspects of the operator are further considered as iterations proceed. Although the iterates are correlated, it would be useful to know their effect on the probability density function of the signals. In other words, although there appears to be little point in using several iterates as features, there may be reason to choose an iterate other than the first one.

In order to investigate this aspect of the work, signals are taken from two common probability density functions, the Gaussian and the Uniform. The aim of the experiments is to gain an insight into the effect of the operator and successive iterates on the statistics of the signal. For example, as was shown earlier there is an increase in power with iterations. This power change is reflected in both the high and low frequency components. The effect on the mean and variance of the resulting signal is now studied. Furthermore, if these parameters change, the change in distribution of the feature signals either in position, shape or both should be examined.

2.2.4. Experiments with signals from different probability distributions.

(a) Signals from a Gaussian distribution.

The two signals shown in figures 2.14a and 2.14b both have a mean of zero and variances of 50 and 70 respectively. Using a λ value of 1 and an operator length of 3, five iterations of the operator were used. For each iteration, the statistical parameters were measured. Figures 2.15a and 2.15b show the probability density functions for each original signal and its first five iterates respectively. Tables 2.5a,b summarise the relevant statistical parameters.

	Iterates					
	Original	**1**	**2**	**3**	**4**	**5**
Mean	-0.11	12.21	17.28	20.89	23.77	26.35
Variance	43.21	34.87	35.20	36.17	37.42	37.68
Skew	-0.04	0.66	0.29	0.15	0.03	-0.07
Kurtosis	0.10	0.14	-0.31	0.34	1.01	1.95

Table 2.5a. Statistics for Gaussian N(0,50) distribution signal and first 5 iterates.

	Iterates					
	Original	**1**	**2**	**3**	**4**	**5**
Mean	-0.14	14.30	20.12	24.17	27.33	30.15
Variance	60.51	48.95	49.20	50.55	52.27	52.42
Skew	-0.04	0.66	0.29	0.17	0.08	-0.01
Kurtosis	0.10	0.13	-0.34	0.30	0.89	1.70

Table 2.5b. Statistics for Gaussian N(0,70) distribution signal and first 5 iterates.

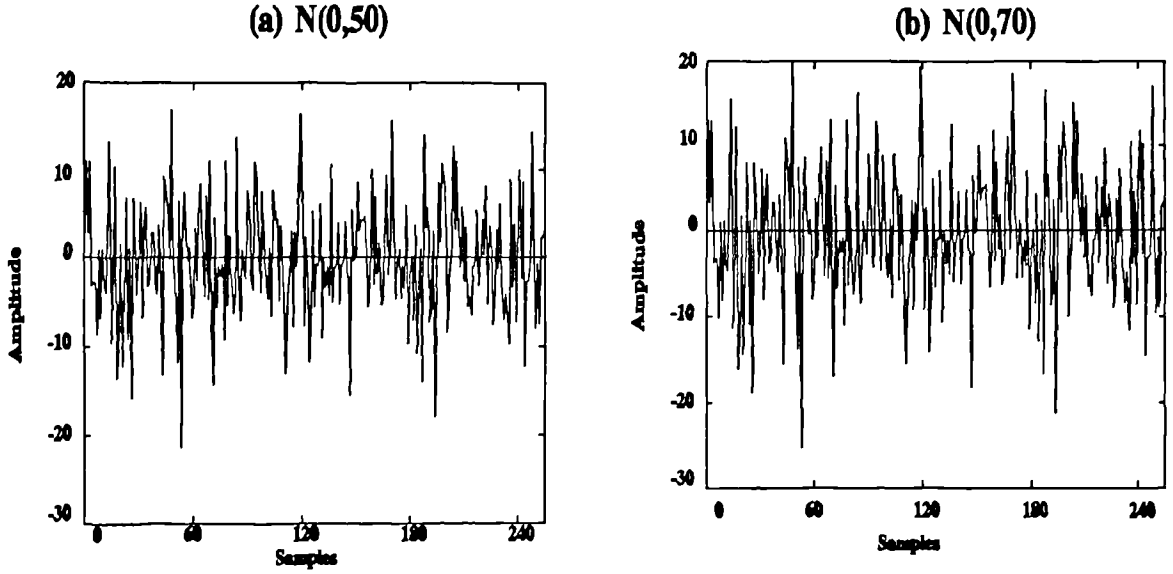


Figure 2.14. Gaussian signals from distributions N(0,50) and N(0,70).

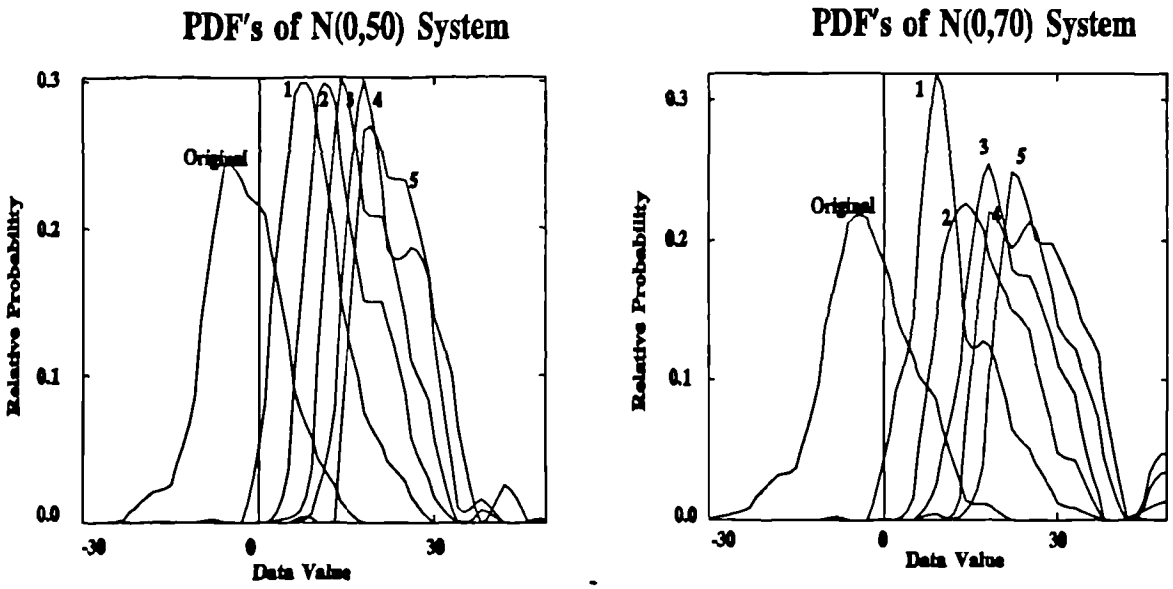


Figure 2.15. Probability densities for above signals and first 5 iterates.

From these results several points emerge.

a). Although in this case both original signals had the same initial mean of zero, it was found that starting with signals of different means produced identical results. The operations are therefore (as expected) independent of the mean of the starting signal. This is important from the point of view of texture segmentation. Since the aim is to segment texture, then segmentation must not separate identical textures that only differ in their mean value. These type of conditions are often seen in images where there is a variation in lighting across a texture.

b). The mean progressively increases with iterations.

c). The variance increases slightly with iterations, but essentially remains constant.

d). The distribution remains symmetrical, but becomes more evenly distributed

e). As the iterations proceed, and as a consequence of (b) and (c) the two distributions become separated.

(b) Signals from a uniform distribution.

The purpose of using another distribution is to investigate how that distribution is affected by the operator and its iterates. Consideration is given to whether uniform density is maintained or Gaussian or some other form obtained. As in the previous section two signals from a uniform distribution were prepared. Both had a mean of zero and variances of 50 and 70, and they are illustrated in figures 2.16a and 2.16b respectively. The

probability density functions for the original and the first five iterates are shown in figures 2.17a and 2.17b respectively. Tables 2.6a,b summarise the statistical parameters.

	Iterates					
	Original	**1**	**2**	**3**	**4**	**5**
Mean	-0.72	11.79	16.71	19.92	22.58	24.91
Variance	48.43	30.58	24.66	20.38	16.77	15.09
Skew	0.14	0.20	-0.42	-0.94	-1.49	-2.16
Kurtosis	-1.23	-0.87	-0.28	1.62	5.53	11.31

Table 2.6a. Statistics for uniform U(0,50) distribution signal and first 5 iterates.

	Iterates					
	Original	**1**	**2**	**3**	**4**	**5**
Mean	-0.85	13.82	19.42	22.98	25.86	28.33
Variance	67.80	42.77	34.47	28.00	22.47	19.56
Skew	0.14	0.20	-0.41	-0.92	-1.49	-2.17
Kurtosis	-1.23	-0.88	-0.34	1.44	5.26	11.28

Table 2.6b. Statistics for uniform U(0,70) distribution signal and first 5 iterates.

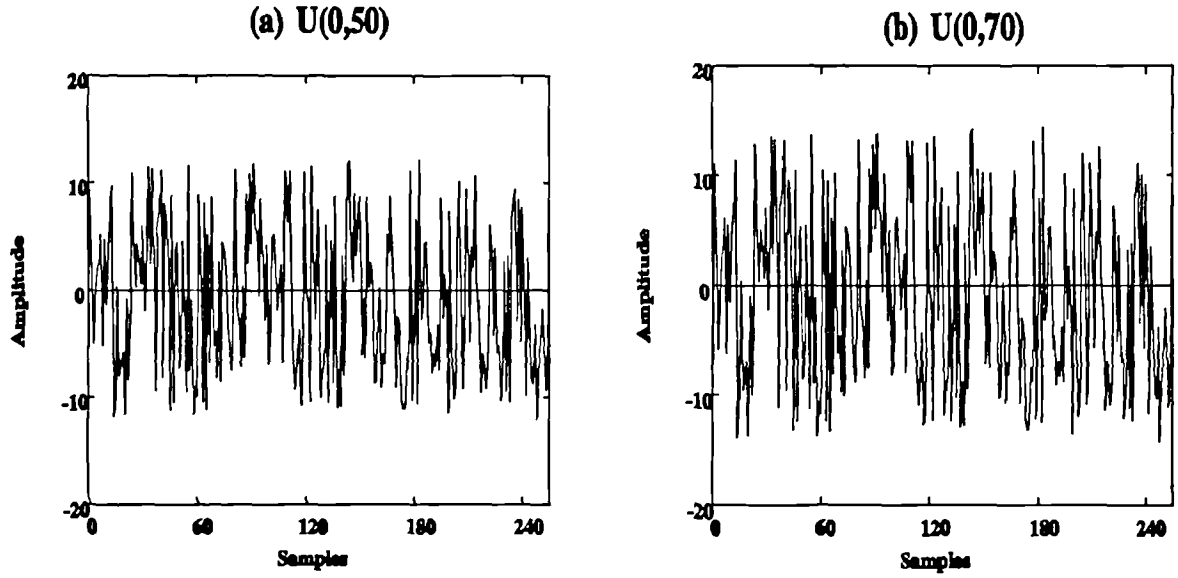


Figure 2.16. Uniform signals from distributions U(0,50) and U(0,70).

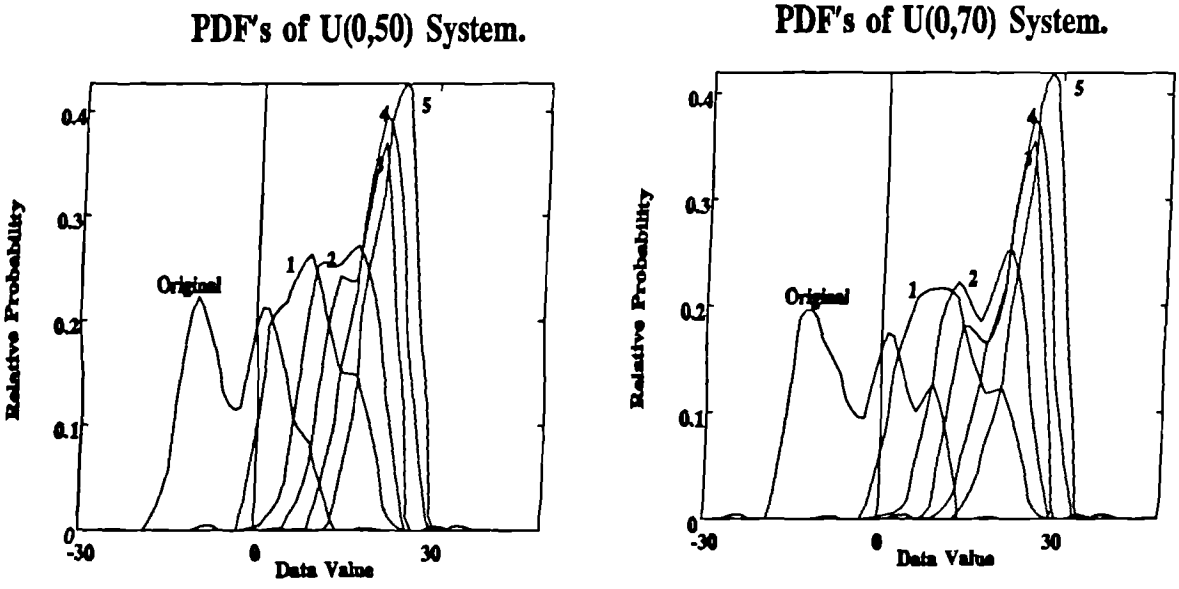


Figure 2.17. Probability densities for above signals and first 5 iterates.

Summarising these results:

a). Although both signals had the same initial mean of zero, it was found that starting with different means produced identical results. The operations are therefore independent of the mean of the starting signal, an important factor as mentioned earlier.

b). The mean progressively increases with iterations.

c). The variance decreases with iterations.

d). The distribution remains almost uniform.

e). Again in this case, as for the Gaussian, the two distributions are being separated as a consequence of (b) and (c). In this case the separation is proceeding more quickly as a result of (c).

Both the Uniform and Gaussian sets of signals have very small differences in variances (50 and 70), the same mean, and yet they are being separated. This can be shown graphically as follows: the two Gaussian signals have been put together side by side as one signal, shown in figure 2.18a. After running three iterations (λ=1) and then performing a sliding *t*-test (window=29) on the result of the third iteration, the results of figure 2.18b are produced. Also shown are the results of the sliding *t*-test on the original signal. The straight line indicates the 0.1% significance level. Figures 2.19a,b show the equivalent results for the uniformly distributed signals. In the sliding *t*-test a window is taken either side of a sample point, the *t*-statistic is then computed from the two windows, large absolute values of the statistic being regarded as significant, the windows are then moved along one sample and the *t*-statistic recomputed. It is these *t*-values which are shown plotted in figures 2.18b,c and 2.19b,c. These show clearly the separation of the two signals at their respective mid-points.

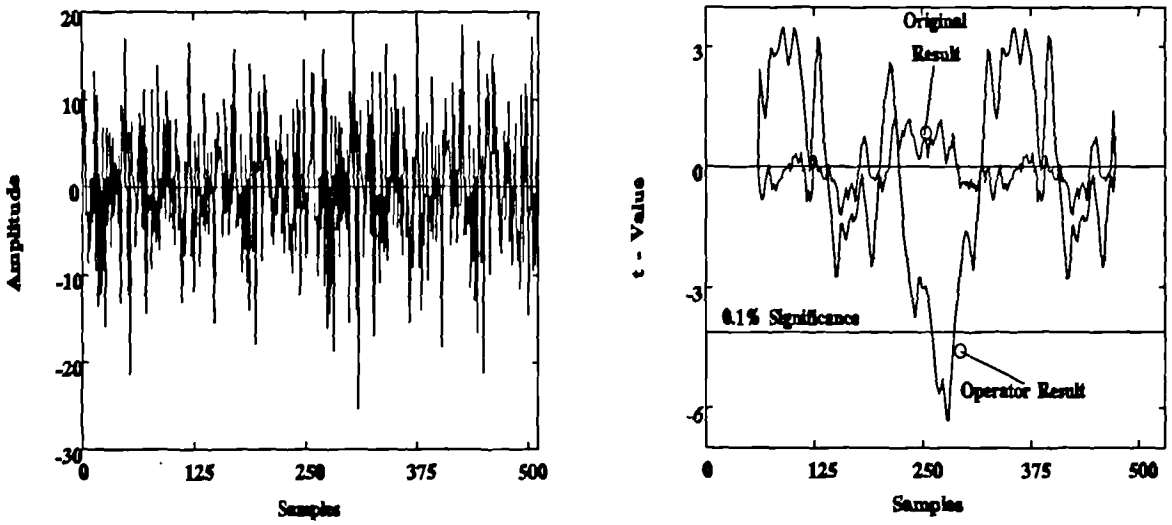


Figure 2.18. Original Gaussian signals(left) and *t*-test results(right).

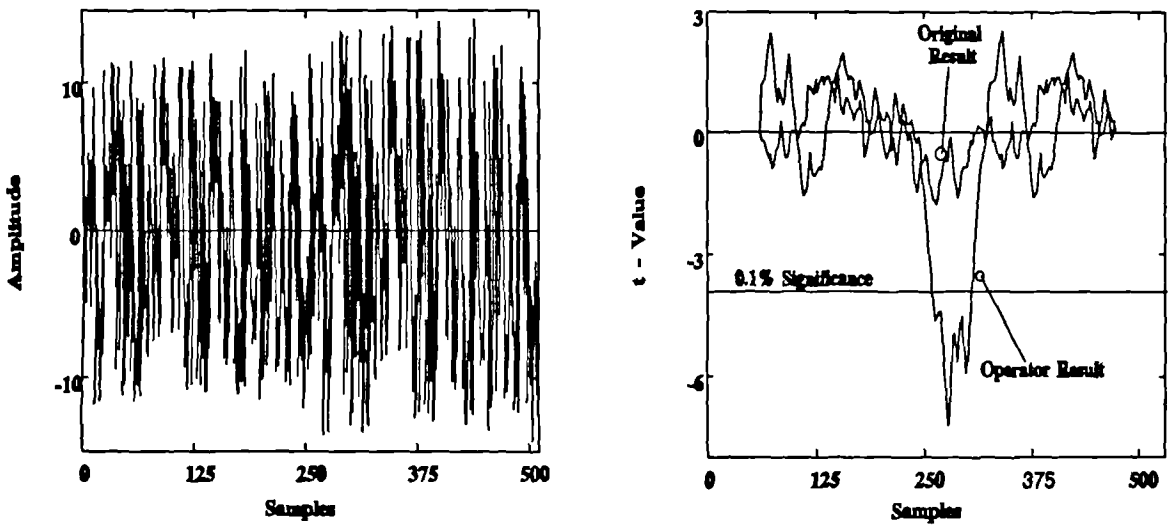


Figure 2.19. Original uniform signals(left) and *t*-test results(right).

The t-statistic is used to measure differences in the means of two populations, and so is a suitable test for this example. Further use will be made of the t - statistic (and its multivariate counterpart) in chapter 4. It is formulated as follows,

$$t = \frac{(\bar{x} - \bar{y})}{s}$$

where

$$s^2 = \frac{\sum x_i^2 - n\bar{x}^2 + \sum y_i^2 - n\bar{y}^2}{n(n-1)}$$

$\bar{x}$, $\bar{y}$ are the means of the windows(populations) and n is the size of the window. x_i, y_i represent the i^{th} samples of the windows.

It is concluded that if the iterates of the fractal measures are to be used as features in a statistical classifier, then only one iterate need be used, and that this has the potential to separate signals. It is now speculated that if such a separation can be achieved in one dimension, ie one direction, then significant separation could be achieved by considering many directions and using the resultant set of features as input to a multi-variable statistical classifier. Furthermore, a separation between two one-dimensional signals to indicate the boundaries has been shown, without knowledge of the boundary location and with the two signals appearing as part of the same signal.

2.3 The use of directionality.

Extending the previous sections work to two dimensions, i.e. images, now allows the incorporation of directionality into the measurement scheme. Just as the operator was defined in one dimension using equations (2.2) and (2.3) for the lower and upper bounds, this same definition now allows the measurement of these bounds in different directions. Figure 2.20 shows the directions based on a 3 * 3 pixel grid and the numbering scheme used to denote each direction. For example, direction 1 is horizontal and uses the centre pixel (which is used for all the directions) and the pixels on its immediate left and right.

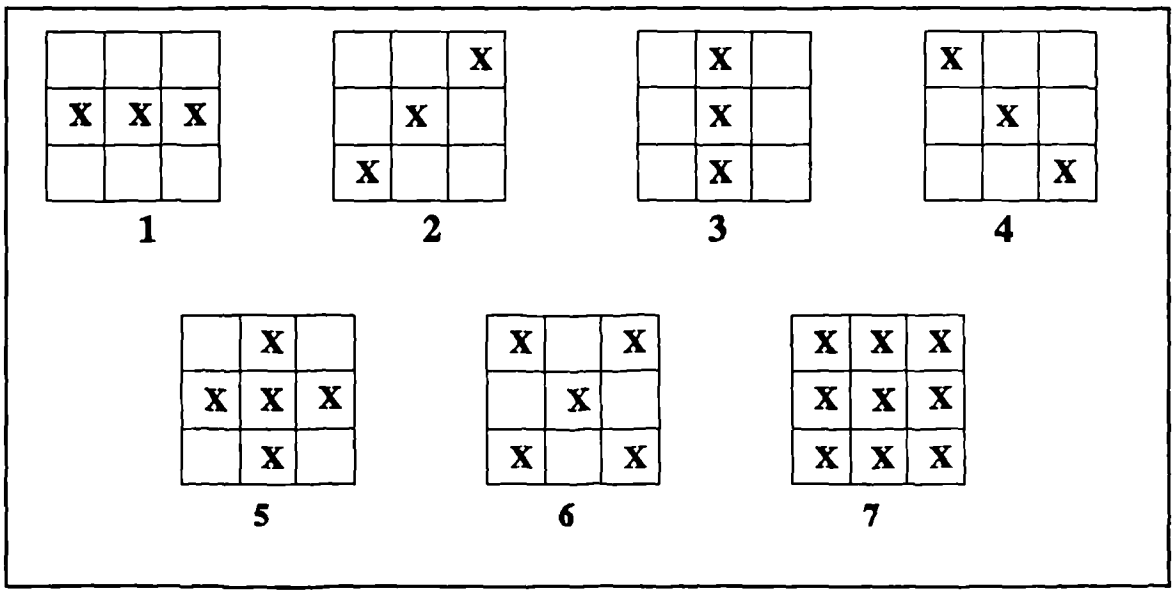


Figure 2.20. Directions used by the operator on a 3 * 3 pixel grid.

To illustrate the scheme for images and texture, a very simple image in terms of texture is used. This has two "texture" fields in it. On the left half of the image are four cycles of a sine wave, where the sine wave runs from left to right of the image. The right hand side of the image is the same sine wave rotated through 90°, that is, it runs from top to bottom. The image is shown in figure 2.21, being 256 * 256 pixels in the range 0..255.

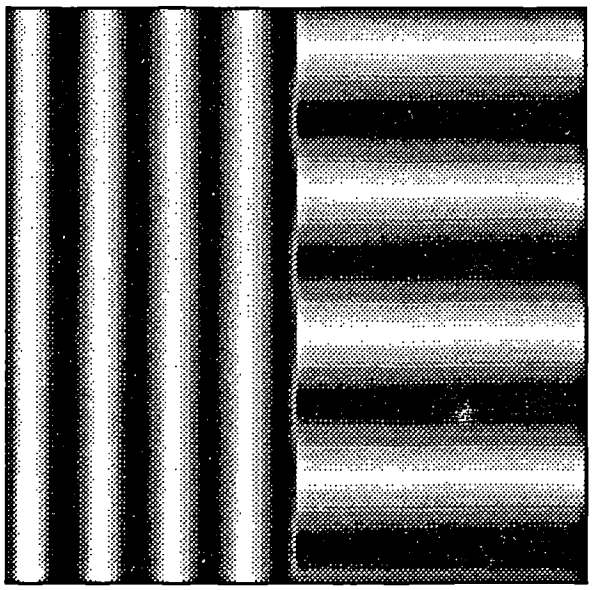


Figure 2.21. Image showing directionality of textures.

It is appreciated that this is a relatively simple image to segment and many schemes will produce a successful segmentation, however it is being used to illustrate the principle of directionality using the present operator.

Because the two main directions are represented by directions 1 and 3 of the operator shown in figure 2.20, then these directions are used in the feature extraction. Of the remaining directions, directions 2 and 4 would produce perfectly correlated features and thus would not be useful together. Directions 5,6 and 7 would produce the same result for each texture field and so again would not be useful. In a real situation of course the directions of the various texture fields in an image would not be known and thus the appropriate directional features to be used would not be known either. This gives rise to the non trivial feature selection problem which is discussed further in chapters 4 and 5.

Having established a two feature problem the operator produces two feature images. For this example it is assumed that a supervised classification scheme is in operation and that samples of the two textures have been obtained for training purposes.

Full details of both supervised and unsupervised classification schemes are presented in chapters 4 and 5 respectively. In this case direction 1 produces a zero feature image for the right hand sine wave and a non zero image for the left hand sine wave. Direction 3 produces the exact opposite. On producing the covariance matrices for these two texture fields, the following are obtained -

$$\begin{bmatrix} 0.00000 & 0.00000 \\ 0.00000 & 0.00324 \end{bmatrix} \qquad \begin{bmatrix} 0.00324 & 0.00000 \\ 0.00000 & 0.00000 \end{bmatrix}$$

Left-Hand Right-Hand

As expected, since the two fields are identical and are right angles to each other, they are totally uncorrelated.

Since neither of these covariance matrices has an inverse, then the quadratic discriminant cannot be used, hence a pooled covariance matrix is formed and a linear discriminant is used. Further discussion of the linear and quadratic discriminants is included in chapter 4. Using the pooled covariance matrix and the mean vectors associated with the features produced by the two directions of the operator, a classification produces a perfect result, successively segmenting the left and right hand texture fields. The two fields are identical and differ only in direction. Since there is a perceived visual difference between the left and right hand sides, then this difference must be highlighted as directionality is an important feature in the present study of seabed textures.

The operator is insensitive to phase. Although the eye perceives these phase differences the operator cannot. To find this change the quadrature signal is used, the quadrature signal being a 90° phase shifted version of the original. This idea is explained

more fully in chapter 3 and has been well researched by Knutsson and Granlund [1.25] and more recently using Gabor filters [1.31], [1.32].

2.4 Conclusions.

In summarising the results of this chapter several important points are highlighted.

The operator in producing a feature image provides an important statistical transform. With each iteration the mean increases and the variance generally decreases in proportion. This is useful for the purpose of separating signals since increasing the mean usually increases the variance in proportion and this does not help classification. This is illustrated in figure 2.22. In this figure the results of the operator on the two Gaussian signals with mean zero and variances 50 and 70 is shown. The graphs show the result of successive iterations on the statistics of the signals. As can be seen there is a strong increase in the mean and a corresponding decrease in the variance as iterations proceed.

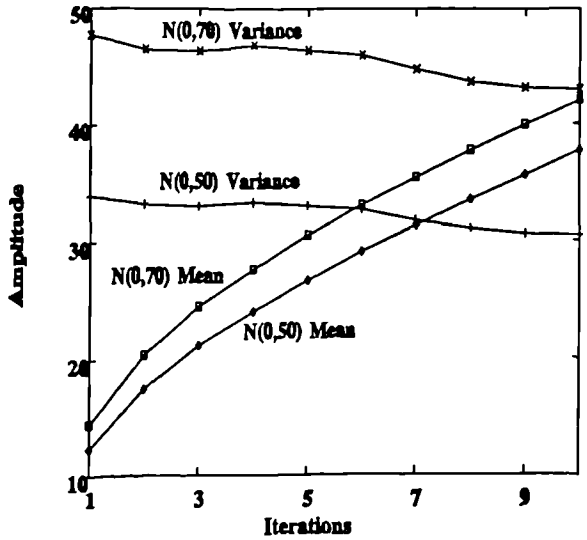


Figure 2.22. Statistics of iterations on Gaussian signals.

Furthermore in selecting a value of λ for the operator it is to be noted that one iteration at level 2λ is not the same as two iterations at level λ. This is highlighted in figure 2.23. Figure 2.23 shows the result of two iterations of the operator using λ=1 on a fractal signal of dimension 1.2, and one iteration at λ=2. Clearly the two results are different. Further the statistics of the two signals are clearly different.

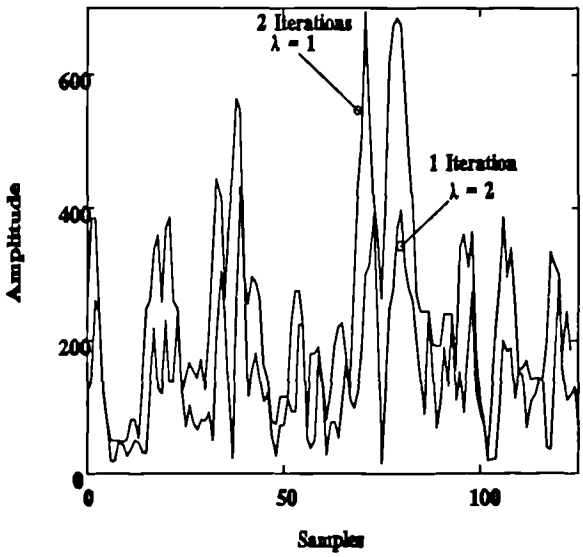


Figure 2.23. Two iterations at λ, and one iteration at 2λ.

The important step is the first iteration producing the upper and lower blanket to produce the difference result. From then on, subsequent iterations act on the upper and lower blankets themselves and not on the original signal. The main purpose of more than one iteration is to adjust the statistics in such a manner that similar signals will have sufficiently different statistical parameters to allow separation.

In using this operator it was found unnecessary to measure the fractal dimension of the signal (or texture). This is useful since the computation of the fractal dimension is

time consuming and possibly subject to numerical errors. Thus only one iteration of the operator is used to produce a feature signal or image. An important step in the segmentation is the subsequent processing of this feature set by appropriate statistical techniques. Thus the operator does not itself produce a segmentation, but is only the first part in the segmentation system. The system used to illustrate directionality and segmentation was a supervised technique. This idea is explored further in chapter 4 and extended to unsupervised segmentation in chapter 5. A block diagram illustrating the supervised segmentation system is shown in figure 2.24.

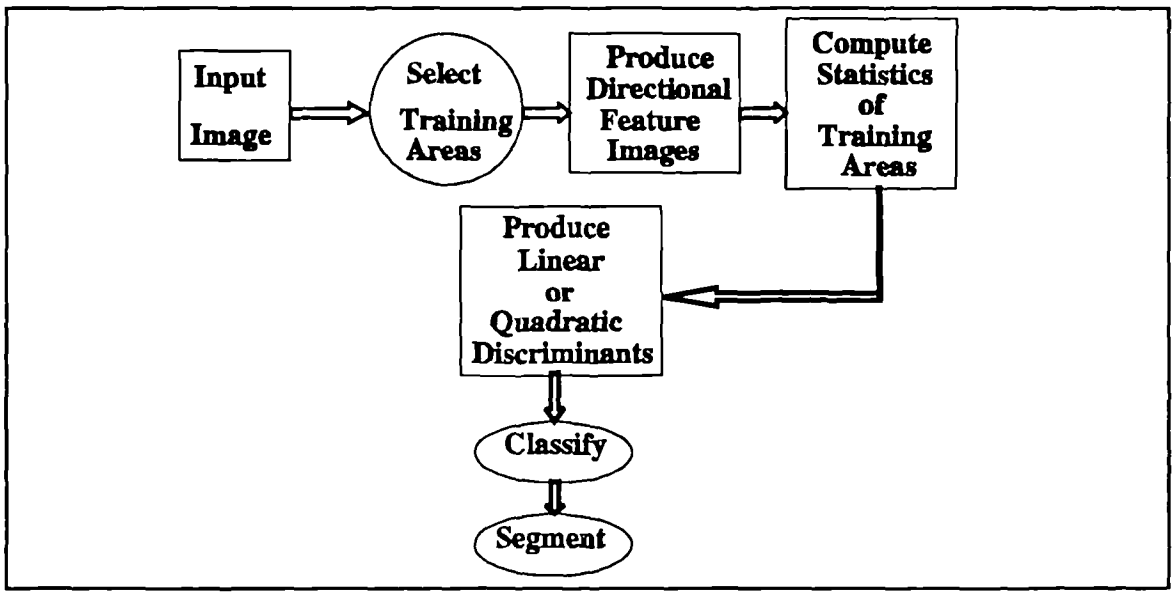


Figure 2.24. Schematic of segmentation operation.

For images of textures the idea of a single number (the fractal dimension) to describe a texture for segmentation purposes should be abandoned. Since directionality is such an important feature in vision, then at the very least, if a fractal dimension is to be assigned then this will require several numbers describing the dimension in each of the dominant directions of the texture. This idea is explored in chapter 3.

2.5. Authors' papers relevant to chapter 2.

1). R.M.Dunbar, I.Hennings, L.Kullander, L.M.Linnett, O.Oskarsson, K.Torlegard, G.Shippey, K.Vikgren, F.Werner, *Monitoring of sea floor changes with multisensors: Problems related to positioning, image rectification and classification*, International Conference on Submarine systems, U-90, Stockholm, Sweden, 7-10 May, 1990, pp17.1-17.58.

2). L.M.Linnett, A.J.Richardson, *Texture segmentation using directional operators*, International Conference on Acoustics, Speech and Signal Processing, Albuquerque, NM, USA, April 3-6 1990, pp2309-2312.

3). L.M.Linnett, A.J.Richardson, *Texture classification using fractal measurements*, Research Memorandum, RM/88/6, Heriot-Watt University, Department of Electrical & Electronic Engineering, 1988.

CHAPTER 3

THE USE OF SPECTRAL MODELLING IN TEXTURE ANALYSIS

3.1. Introduction.

In this chapter it is intended to present a process for the synthesis of textures based on fractal ideas. Since this is a spectral approach, some consideration is given to the influence of phase in textures. Individual textures are produced and the problem of their incorporation into multi-textured images with random boundaries is addressed.

This study of texture analysis includes texture synthesis for the following reasons -

(a) To provide a source of sufficiently complex and realistic data for analysis.

(b) To provide *ground truthed* data in the sense of having known statistical and spectral information.

(c) To provide a greater insight into the strengths and weaknesses of the texture analysis algorithms.

(d) To provide further knowledge in the understanding of texture.

The main application of the present study is in the classification and segmentation of seabed textures. The acquisition of sufficient data of good quality is difficult to achieve. Expert analysis by skilled sonar interpreters and geologists is often necessary to determine the natures of the textures contained in the data. In order to experiment with

algorithms, methods should be available that allow small changes in texture characteristics to be made. The algorithm can then be fully tested. Finally, any scheme which produces realistic images of texture should provide a basis for obtaining a greater understanding of texture and its perception by the human vision system.

3.2. The synthesis of texture images.

The advent of fractals [1.40] and the impetus given to imagery by computer graphics has resulted in the creation of many realistic images. Some excellent examples of computer generated fractal images of natural scenes have appeared. These began with the work of Voss [1.41] and have appeared as illustrations in the work of Mandelbrot [1.40]. They are mainly of mountainous terrain and, by careful control of colour mapping, clouds. A realistic planet can be produced by using computer graphics techniques which enable a surface to be wrapped around a sphere. Some examples of these are shown in figure 3.1.

Since then other authors have extended the work. In particular the works of Lewis [3.1] and Mastin [3.2] are particularly impressive. Lewis used stochastic techniques for modelling a texture based on a given autocorrelation function, and Mastin et al. produced a sea model using an intricate formula based on the Pierson-Moskowics spectrum [3.3]. Neither of these techniques was fractal based.

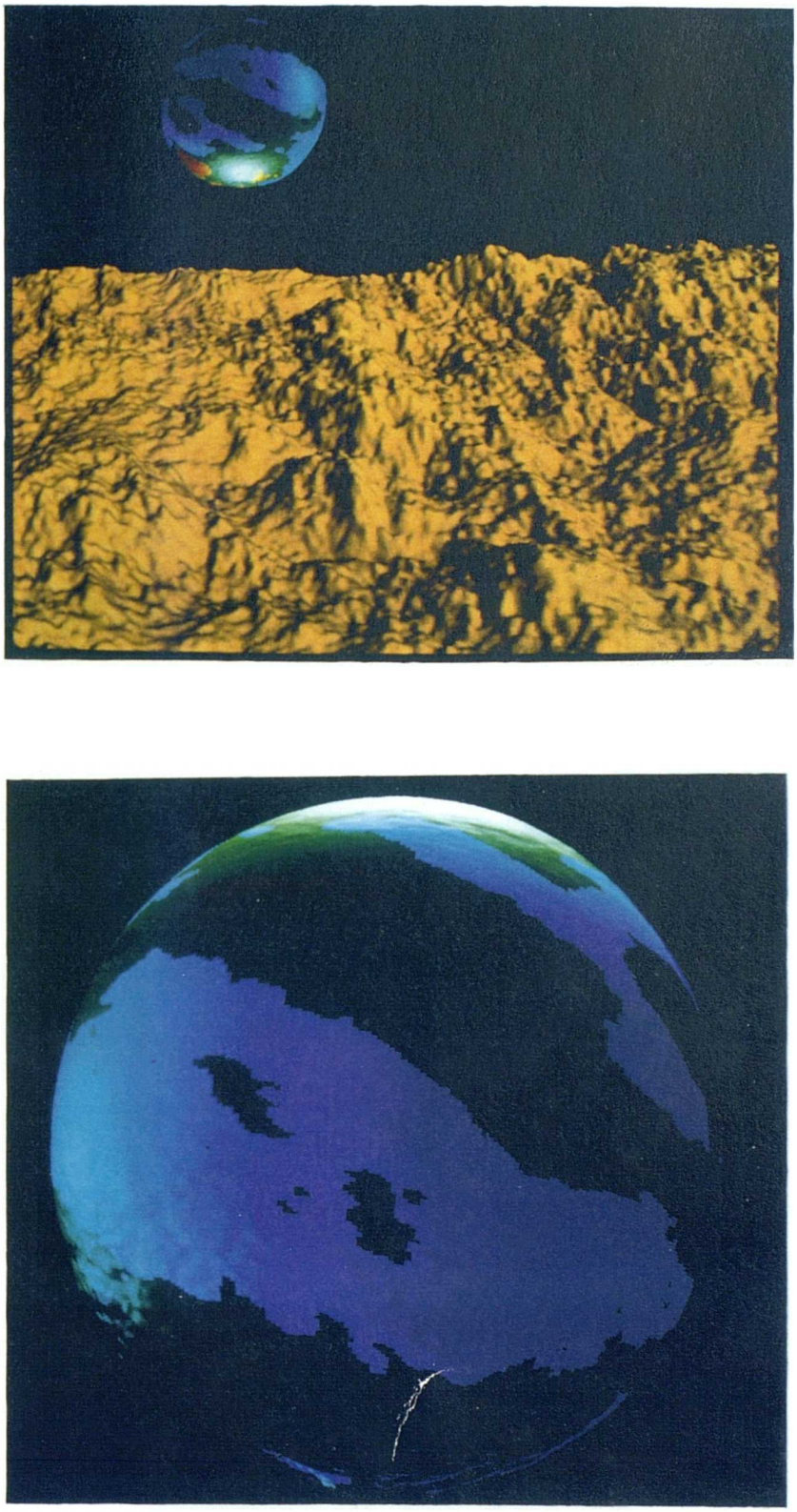


Figure 3.1. Examples of fractal images.

The majority of such images are generated to look like natural images but not in imitation of a specific scenario. There has been no necessity to control the precise detail in the image. One of the aims of the work described in this chapter is to produce realism in the images generated, but equally important is to have the facility to generate textures within prescribed boundaries.

In this work emphasis is solely on modelling in the frequency domain in two dimensions. In essence it is similar to designing suitable 2D filters for filtering a random noise image. Variations on the fractal model are made i.e. spectra with characteristics significantly different to the traditional $1/f$ shape are produced.

3.2.1 The shape of the spectral density function.

For simplicity the aspects of the model in 1D will be shown. From this the extension to 2D is straightforward. Voss [1.41] showed how the fractal dimension was related to the power spectral density as :-

$$D = E + \frac{(3 - \beta)}{2} \tag{3.1}$$

or in terms of amplitude spectra,

$$D = E + \frac{(3 - 2\beta)}{2}$$

where D = fractal dimension, $E < D < E+1$

E = topological dimension, E = 1 for 1D, 2 for 2D (e.g. landscapes), 3 for 3D (e.g. clouds).

β = exponent of f.

f = frequency.

Assume that:-

$$S \propto \frac{1}{f^{\beta}}$$

where S = power spectral density.

Then when $\beta = 0$ white noise is produced, when $\beta = 1$ then the plot of $\log(S.f)$ versus $\log(f)$ falls off as $1/f$. At $\beta = 2$ the plot is that of Brownian or $1/f^2$ noise. Plots of these functions are shown in figure 3.2.

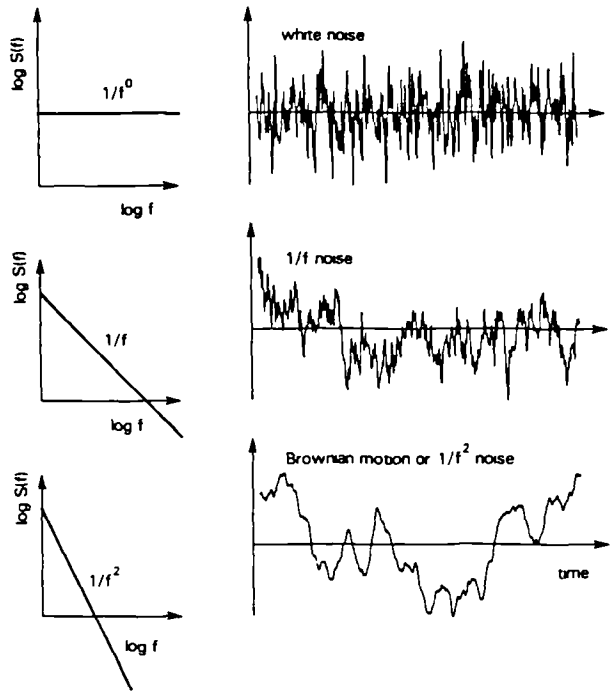


Figure 3.2. Examples of $1/f$ type noise.

In terms of amplitude spectra,

$$A \propto \frac{1}{f^{\beta/2}}$$

where A = spectral amplitude.

These functions must be modelled in the two dimensional frequency domain, to provide suitable models for the texture, based on the power spectral density or amplitude spectrum. The resulting spectra are then inverse Fourier transformed to obtain the texture image in the spatial domain. This has merely modelled the amplitude (or magnitude) of the frequency components, and the phase of these components has not been taken into account. Consider the following -

$$X(t) = A\cos(\omega t + \theta)$$

where A and ω are constants and θ is a random variable uniformly distributed over a range 2π. That is,

$$f(\theta) = \begin{cases} 0 & -\infty < \theta < 0 \\ \dfrac{1}{2\pi} & 0 \le \theta \le 2\pi \\ 0 & 2\pi < \theta \le \infty \end{cases}$$

Then

$$\begin{aligned} R_X(\tau) &= E\,[A\cos(\omega t_1 + \theta)\, A\cos(\omega t_1 + \omega\tau + \theta)] \\ &= E\left[\frac{A^2}{2}\cos(2\omega t_1 + \omega\tau + 2\theta) + \frac{A^2}{2}\cos\omega\tau\right] \\ &= \frac{A^2}{2}\int_0^{2\pi}\frac{1}{2\pi}[\cos(2\omega t_1 + \omega\tau + 2\theta) + \cos\omega\tau]\, d\theta \\ &= \frac{A^2}{2}\cos\omega\tau \end{aligned}$$

Where $R_X(\tau)$ represents the autocorrelation function, which is independent of the phase θ.

Since the autocorrelation function is related to the power spectral density by the Wiener-Khinchine relationship, as follows -

$$S_X(\omega) = \int_{-\infty}^{\infty} R_X(\tau) e^{-j\omega\tau}\, d\tau$$
$$= \mathcal{F}[R_X(\tau)]$$

the power spectral density and hence the amplitude spectrum are also independent of the phase. All of the models used are generally based on having a uniform random phase distribution from $-\pi$ to π. Variations on this will be considered when the phase aspects are considered in greater detail in section 3.3.

The essence of modelling using fractal ideas is that the power spectral density obeys a $1/f$ relationship. Burrough [1.42] found that many natural phenomena have characteristic fractal dimensions, e.g.clouds, craters, shorelines and wind-blown sand. In having this characteristic fractal dimension, they have associated power spectral densities in which the fractal dimension is related to the frequency decay exponent, β, by equation (3.1).

To facilitate development of the models, the spectra of several natural textures were examined. In presenting the spectral data the following conventions have been observed :-

(1). All computations using the two dimensional fast Fourier transform (FFT) (whether dealing with real and imaginary pairs or magnitude and phase pairs) were performed using the data in an *unswapped* format.

(2). The figures showing the pictorial presentation of the spectra are shown in a *swapped* format.

Essentially moving from swapped to unswapped (or vice-versa) involves a translation through two quadrants. Figure 3.3 shows the presentation schematically.

On the left of figure 3.3 is the representation of the unswapped format and on the right, the swapped format. The pictorial representations thus have the zero frequency component(DC) at the centre of the plot.

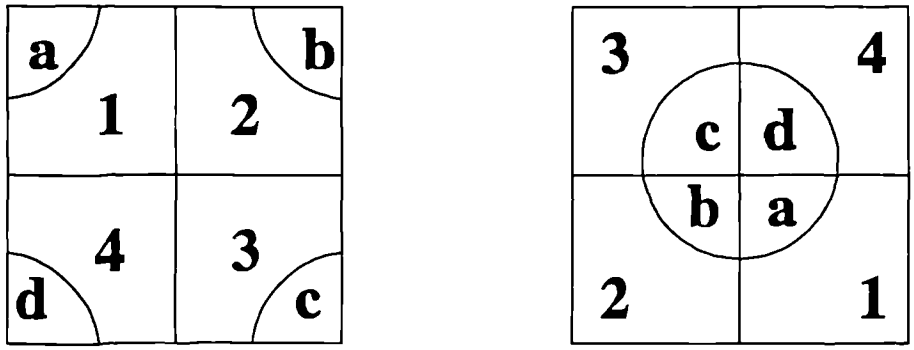


Figure 3.3. Schematic representation of 2D Fourier transform formats.

To aid the interpretation of the spectral data, polar plots were also used. In this representation, the amplitude spectrum, $A(f)$, is transformed to $A(f,\theta)$ where f represents radial frequency and θ represents direction. Thus for each direction (θ), the amplitude spectrum $A_\theta(f)$ may be found, and similarly for each frequency $A_f(\theta)$ may be found. The two forms may be represented as -

$$A(f) = \sum_{\theta=0}^{\theta=\pi} A_\theta(f)$$

and

$$A(\theta) = \sum_{f=1}^{R} A_f(\theta)$$

where for an $N * N$ image, R would usually be $N/2$.

Figures 3.4 to 3.7 inclusive show some natural textures and their respective spectra in various formats.

Figure 3.4 is a seaweed texture, figure 3.5 is an image of cork texture, figure 3.6 is an image of fur texture and figure 3.7 is an image of a water surface.

From these figures the following points emerge.

a) Each has a characteristic set of spectra.

b) Some have dominant frequencies and directions, while others have no preferred directions or dominant frequency (other than DC).

c) Each has varying degrees of amplitude roll-off from the peak frequency.

It is apparent from these figures that there is no simple model which could be used to generate such detail. It is because of this that the fractal model is tried. It is relatively simple and does have a precedent of successfully modelling natural phenomena. Figures 3.4 to 3.7 (inclusive) which show the complexity of the textures in the frequency domain (in varying display formats) are no assistance in respect of phase. This aspect is considered in section 3.3.

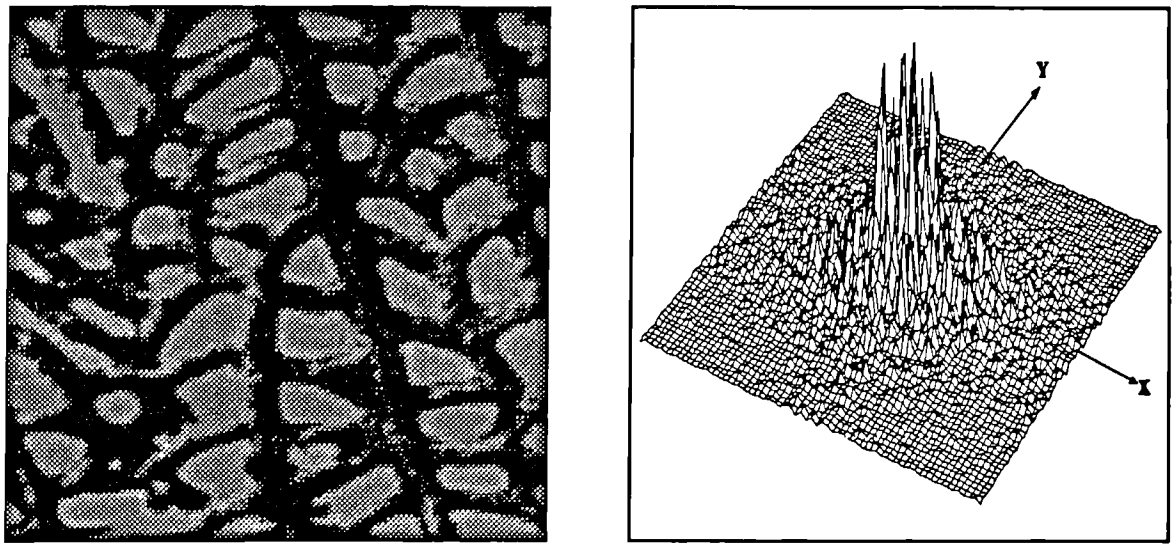


Figure 3.4. Seaweed texture and its pseudo 3D Fourier transform plot.

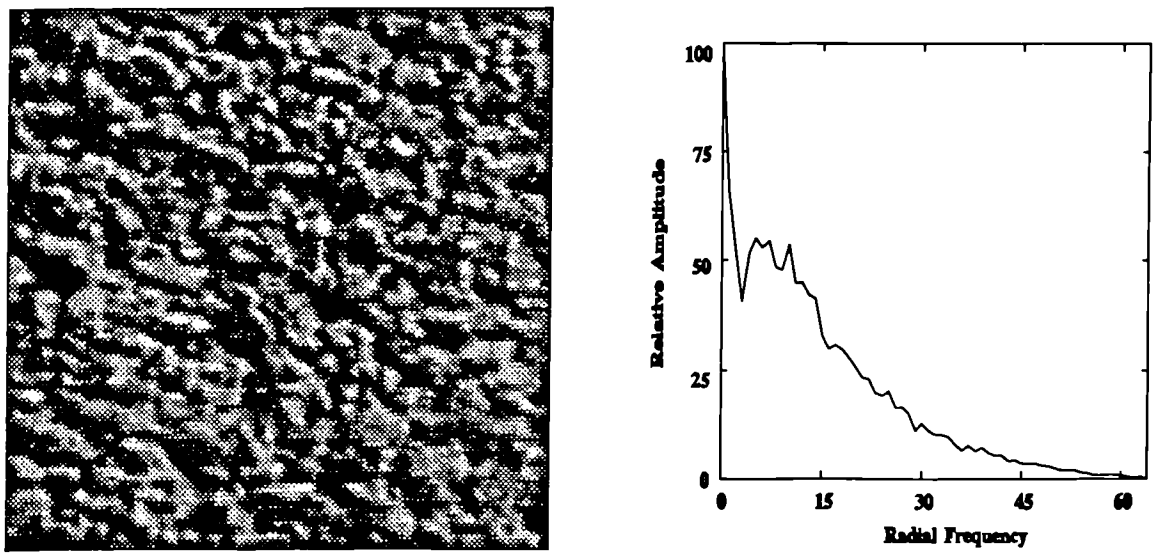


Figure 3.5. Cork texture and its radial frequency plot.

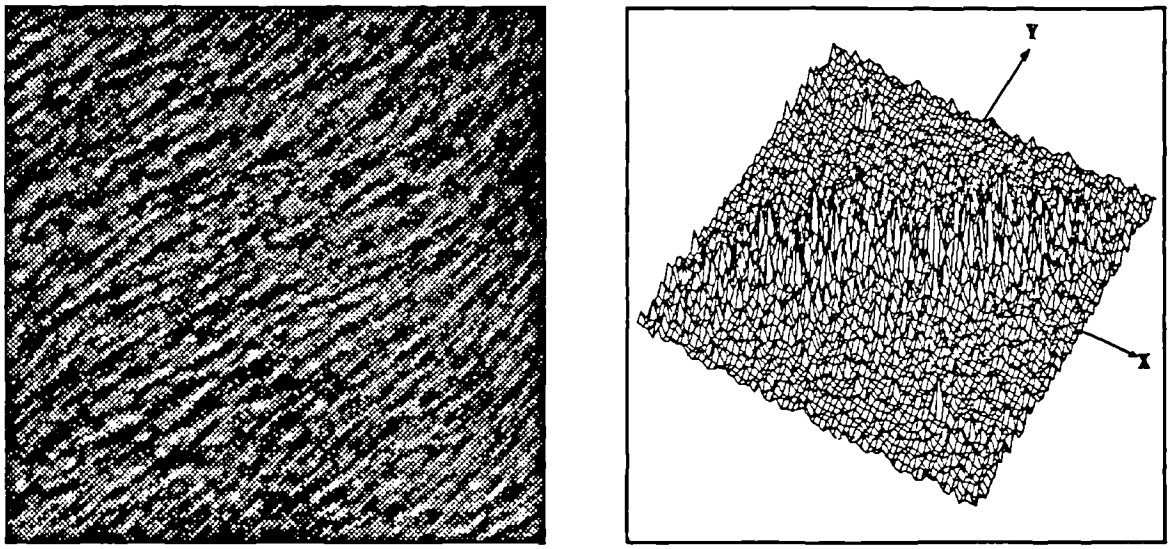


Figure 3.6 Fur texture and its pseudo 3D Fourier transform plot.

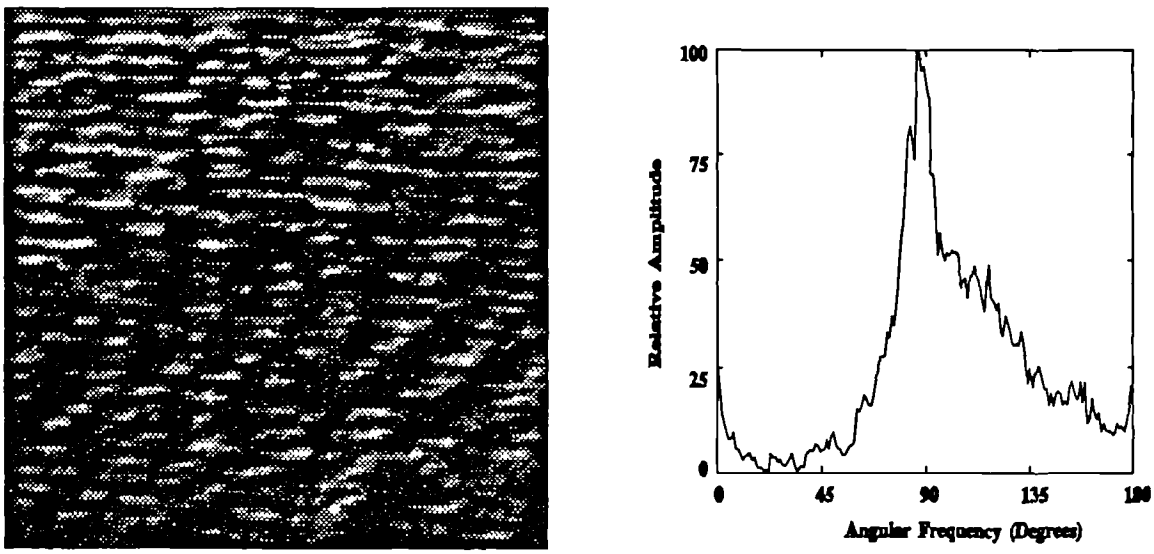


Figure 3.7. Water surface and its angular frequency plot.

Consideration is now given to a set of models that have been created to provide a test bed for testing segmentation algorithms and for producing realistic images.

3.2.2. Some fractal models of texture.

(a) Model 1.

This is the simplest model and is the one which directly models the fractal processes used to generate the examples shown in figure 3.1. In this process only the frequency decay parameter, β, is allowed to vary. The peak frequency is fixed at zero(DC) and the roll off from the peak frequency is the same in all directions, i.e. it is isotropic. This is reflected in the fact that there is a lack of any dominant frequency or direction in the images. Figures 3.8a and 3.8b show the spectral plot and the resulting image respectively for a value of $\beta = 1.3$. The phase is uniformly random in the range $-\pi$ to π. From this simple model, it may be observed by a visual inspection of figures 3.8b, 3.8c, and 3.8d that figure 3.8b is the more natural to the eye as texture. Figures 3.8c and 3.8d are too "noisy". This is reflected in the β values for these three images,

Figure 3.8b, $\beta = 1.3$, Fractal Dimension, D = 2.2

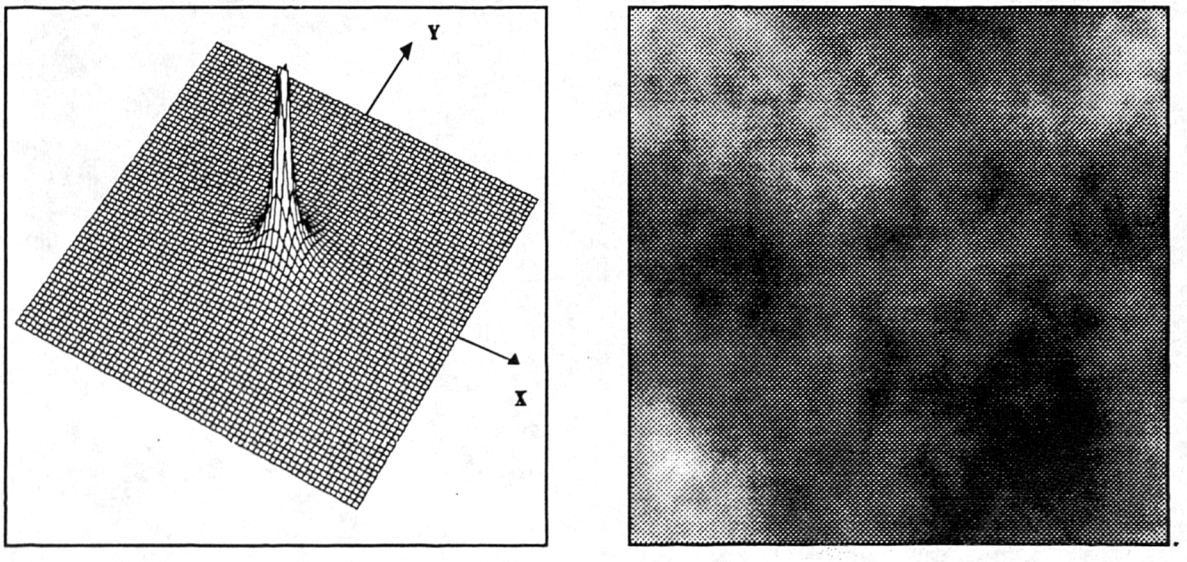
Figure 3.8c, $\beta = 0.9$, Fractal Dimension, D = 2.6

Figure 3.8d, $\beta = 0.7$, Fractal Dimension, D = 2.8

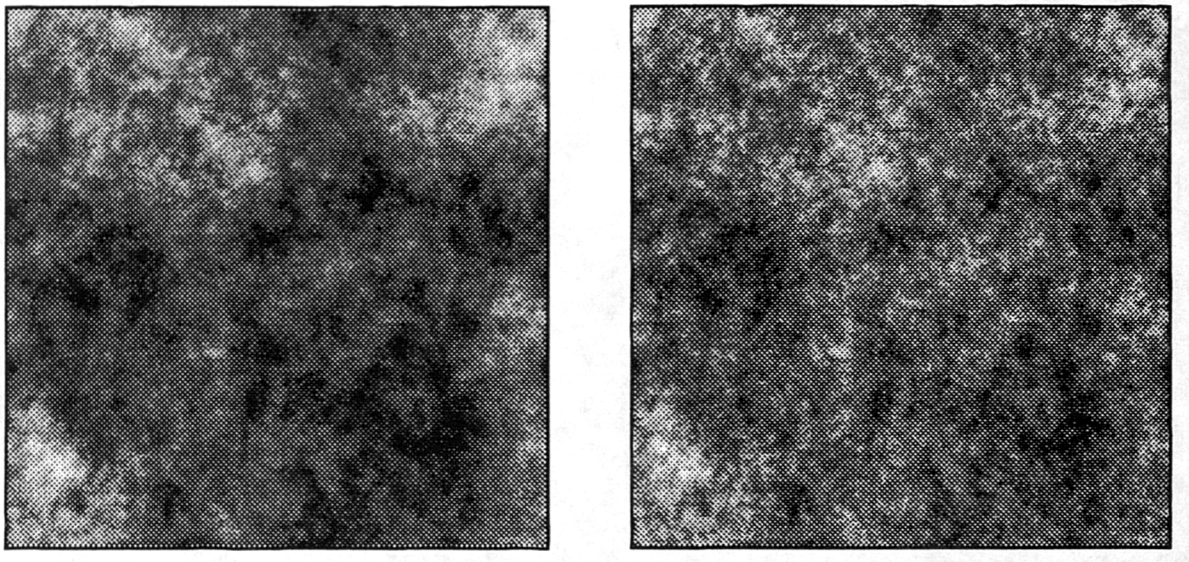
Wher the amplitude version of equation (3.1) has been used.

The fact that the most "natural" image has a fractal dimension of 2.2 further reinforces the work of Burrough [1.42] and others [3.4],[3.5] who have shown that many natural phenomena have fractal dimensions around 2.2. Thus from this model the roll off of the frequency components is seen to be important and may be restricted to a narrow range around β=1.3 in order to produce an image which is at all realistic to the human observer.

The type of image produced by this model is capable of producing only a limited texture range but has a contribution to make when considering shading (section 3.2.3 refers).



Figures 3.8a,b. Spectral plot(left), image(right), fractal dimension, 2.2



Figures 3.8c,d. Model 1 textures, fractal dimensions 2.6(left), 2.8(right).

(b) Model 2.

The next stage in using the fractal modelling technique is to maintain the isotropic decrease of the frequency decay and to allow the peak frequency to move away from DC. Immediately many variations present themselves. The simplest is to move the DC component to a position within one of the quadrants (duplicating it in the diagonally opposite quadrant, to preserve the symmetry relationships of the Fourier Transform) and observing the effect. The result is now an image which is very similar to some textures found naturally on the sea floor. Figure 3.9 is used to illustrate this model. Figure 3.9a shows the spectral plot and figure 3.9b shows the image produced from this with a β value of 1.4 (giving D=2.1). In this image the peak frequency in the x-direction (f_x) was 8 units and that in the y-direction (f_y) was 4 units. The image size is 256 * 256 and the range of pixel values is 0..255. This model then simply locates a peak frequency(F_{peak}) given by :-

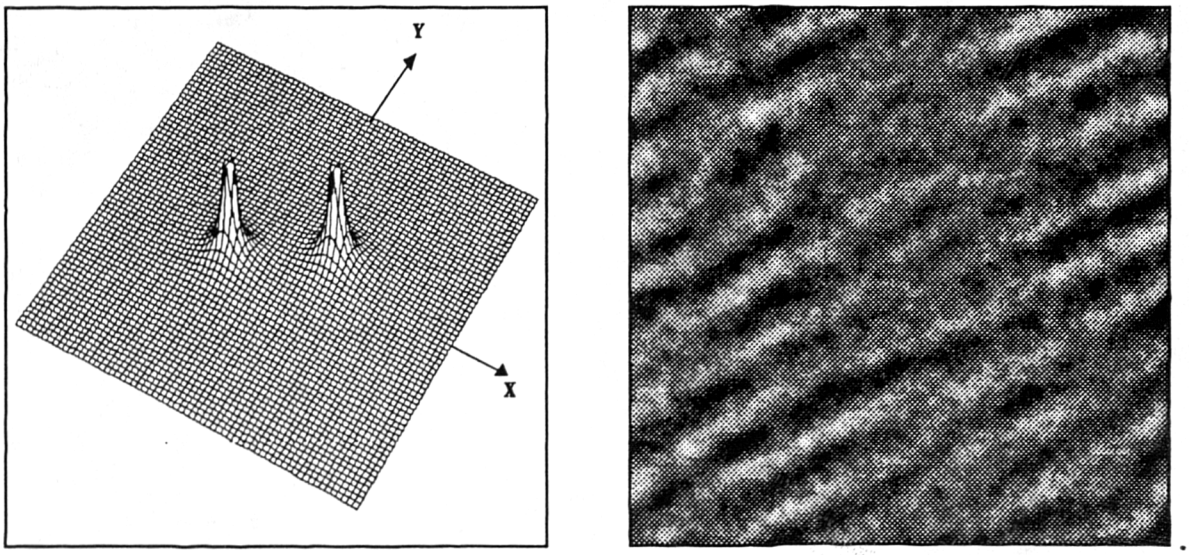
$$F_{peak} = \sqrt{f_x^2 + f_y^2}$$

where the direction(θ) of the texture is given by :-

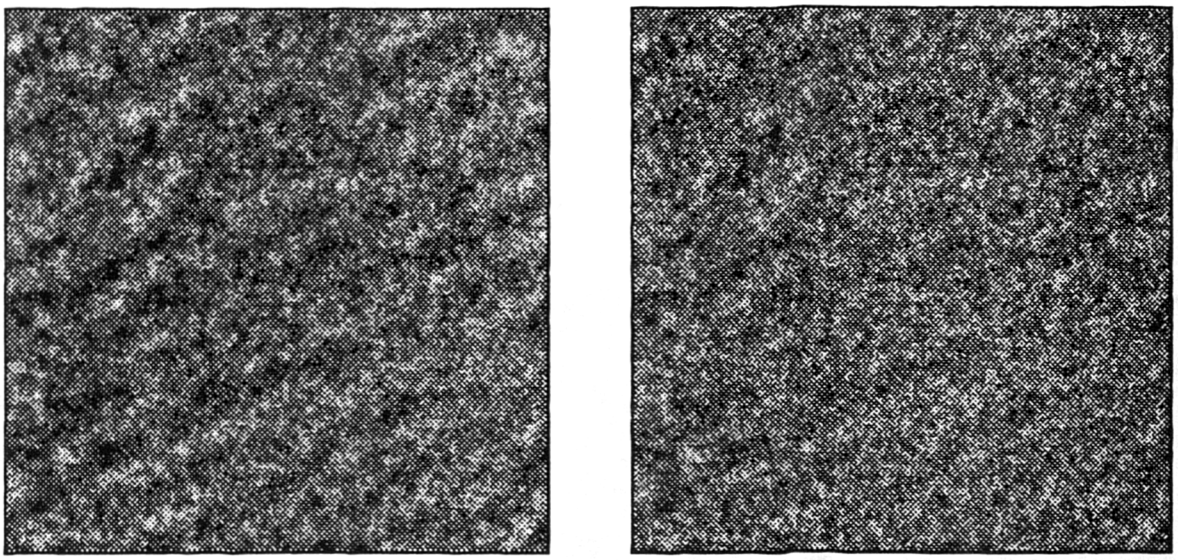
$$\theta = \tan^{-1}\left(\frac{f_y}{f_x}\right)$$

The frequencies away from the peak are evaluated as $1/(F^{\beta})$ where F is found from an expression similar to that for F_{peak} at the particular x and y frequencies. For completeness figures 3.9c and 3.9d illustrate the effect of moving β away from the "natural" value of

2.2. Figure 3.9c has β=0.9, and figure 3.9d has β=0.7. Again these show that for these models too much high frequency content has been introduced and the result is less pleasing to the eye as a natural looking texture.



Figures 3.9a,b. Model 2, spectral plot(left), image(right), β = 1.4

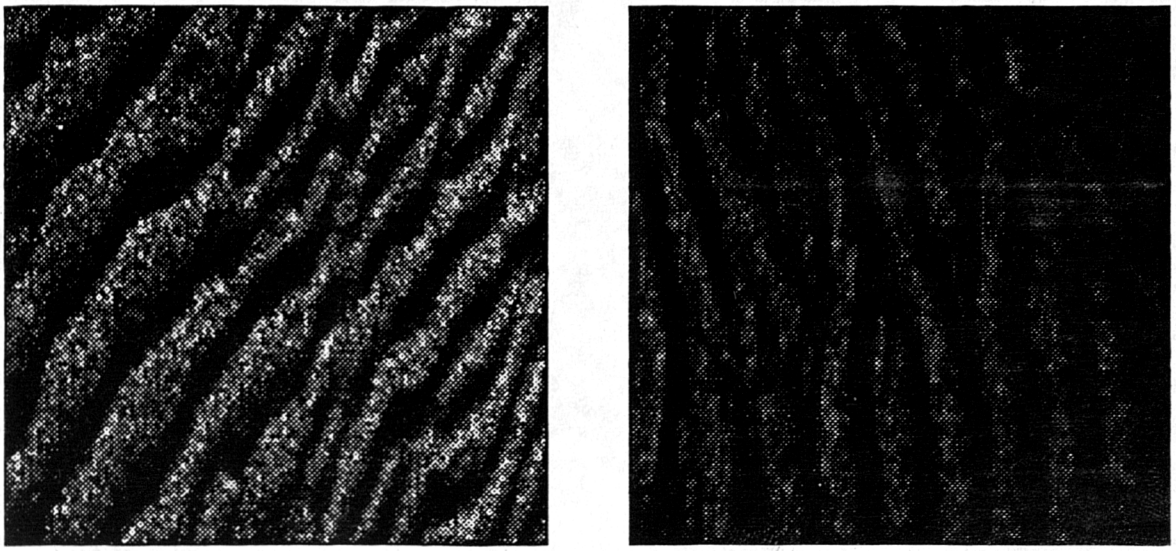


Figures 3.9c,d. Model 2 textures, β = 0.9(left), β = 0.7(right).

For comparison with this model figures 3.10a and 3.10b show two sonar images of the sea bed. These were taken with different sonar instruments and from completely different geographical locations (the north-east coast of the USA and the South coast of England). The visual likeness between the model and reality is significant.

If this model is extended to include more peak frequencies, more images of plausible textures emerge. For example, if the peak frequency of Figure 3.9 is duplicated in another quadrant, now giving the same peak frequency in each of the four quadrants, a more structured texture appears as is shown in figure 3.11a. In the synthetic image f_x = 8 units and f_y = 4 units were used with β=1.4, and phase uniformly random, $-\pi..\pi$.

The idea of adding more peak frequencies may be extended, producing in effect multifractal textures. If a second peak frequency is added to the one shown in figure 3.9, the results shown in figure 3.11b are obtained, again showing a significant likeness to a natural seabed texture.



Figures 3.10a,b. Sonar texture images.

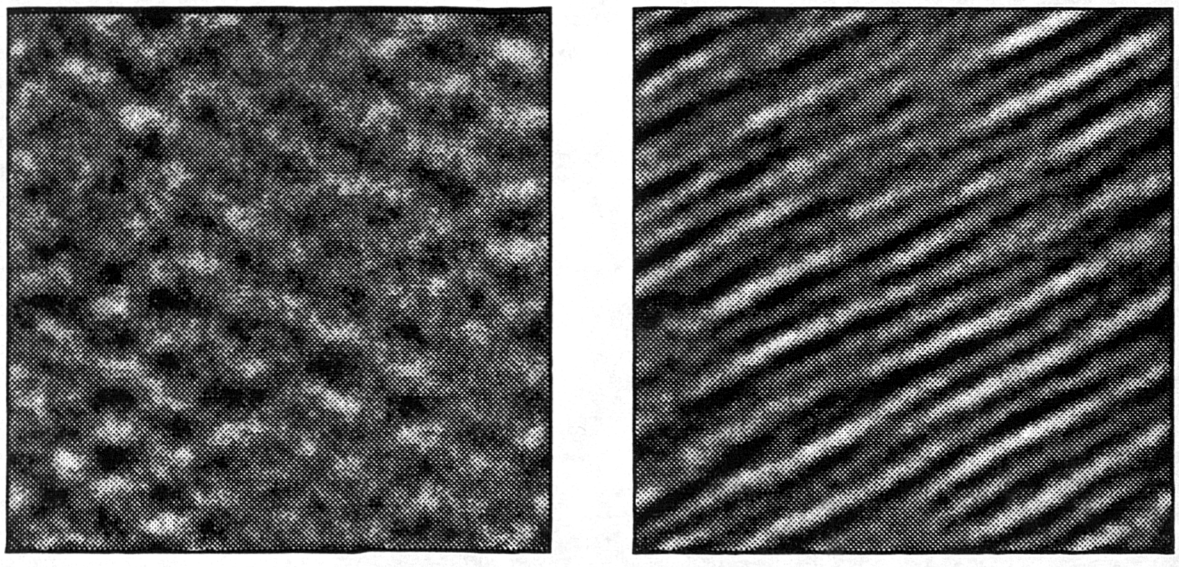


Figure 3.11(a)(left), (b)(right). Model 2, multifractal texture images.

(c) **Model 3.**

This model is similar to model 2 except that the frequency decay is anisotropic. The amplitude of the frequency response will be allowed to decay in a $1/f^{\beta}$ manner in a direction from the peak frequency to DC and in a similar manner at right angles to this. The value of β in each direction need not be the same. This model was developed to investigate the effect of anisotropy on the texture, and whether or not differing roll off rates in different directions affected the visual appearance of the texture. Mathematically the modelling equations are :-

$$F_{xy} = \frac{1}{|f_x - f_{xpeak}|^{\beta_x}} \cdot \frac{1}{|f_y - f_{ypeak}|^{\beta_y}}$$

$$\text{where } |f_x - f_{xpeak}| = 1 \quad \text{if } |f_x - f_{xpeak}| \leq 1$$

$$\text{and } |f_y - f_{ypeak}| = 1 \quad \text{if } |f_y - f_{ypeak}| \leq 1$$

Here $|\ |$ signifies absolute value, β_x and β_y are the roll offs in the x and y directions, f_{xpeak} and f_{ypeak} are the peak frequencies in these directions, and f_x, f_y are the frequencies at a point (x,y) in the frequency domain.

A frequency plot based on this model is shown in figure 3.12a, showing clearly the directional decay of the frequency response. Figure 3.12b is the resulting image produced by inverse Fourier transformation of this plot. The parameters chosen for this image are identical to the parameters chosen for the image in figure 3.9b allowing a visual comparison to be made as to the effects of anisotropy on the textures.

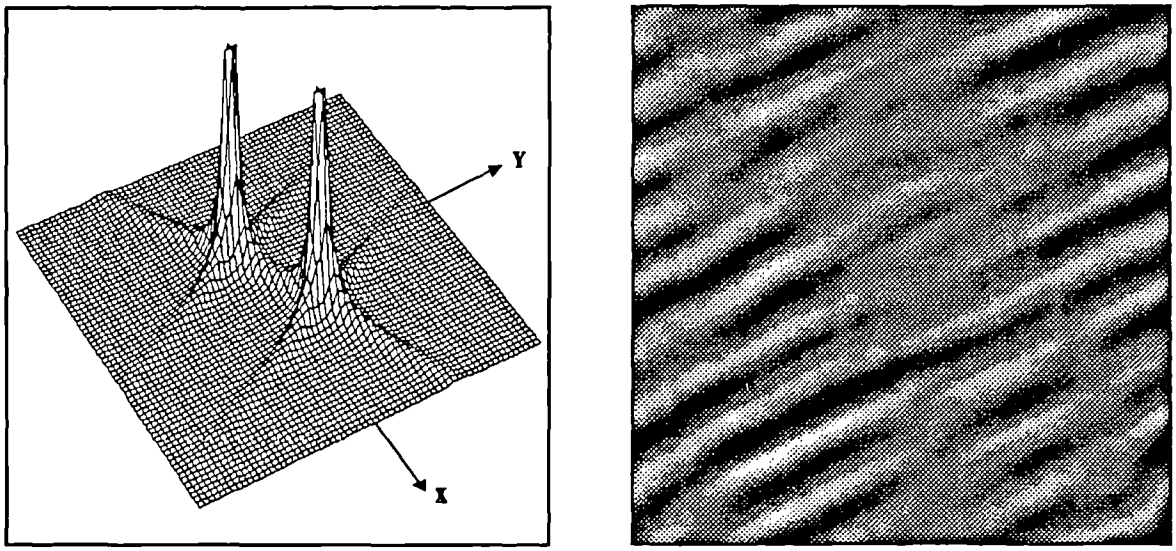


Figure 3.12. Model 3, (a) spectral plot(left), (b) texture image(right).

(d) Model 4.

This returns to a more conventional type of spectrum, - a circularly symmetric narrow band pass spectrum, centred around DC. Mathematically, this is defined as :-

$$r = \left| \sqrt{f_x^2 + f_y^2} - \sqrt{f_{xpeak}^2 + f_{ypeak}^2} \right|$$

whence,

$$F_{xy} = 1 \qquad \text{if } r \leq 1$$

$$= \frac{1}{r^{\beta}} \qquad \text{otherwise}$$

where F_{xy} is the frequency at (x,y).

The purpose in developing this model was to introduce some spatial structure into the texture synthesis while still maintaining the random phase aspect. Figure 3.13a shows a spectral plot for this model, and figure 3.13b shows the resulting image. Again with this

model, it was found that the value of β is restricted to around 1.4 in order to produce visually pleasing images.

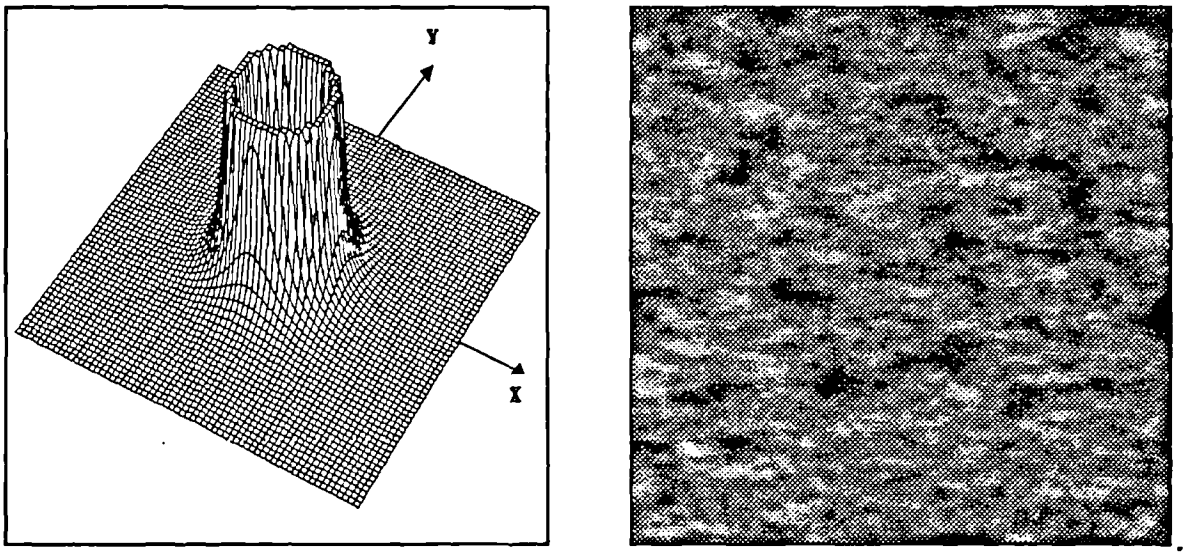


Figure 3.13. Model 4, (a) spectral plot(left), (b) texture image(right).

(e) Model 5.

Model 5 is again conventional, being an elliptical or circular low pass filter, where the roll off from the cut off frequencies obeys a $1/f^{\beta}$ relationship. The form of the filter may be expressed as :-

$$r = \left(\frac{f_x}{f_{xpeak}}\right)^2 + \left(\frac{f_y}{f_{ypeak}}\right)^2$$

Whence,

$$F_{xy} = 1 \qquad \text{if } r \leq 1$$

$$= \frac{1}{r^{\beta}} \qquad \text{otherwise}$$

Figure 3.14a is an example of the spectral plot produced from this model with $\beta = 1.4$, $f_x = 8$ units, and $f_y = 4$ units. The corresponding spatial image is shown in figure 3.14b.

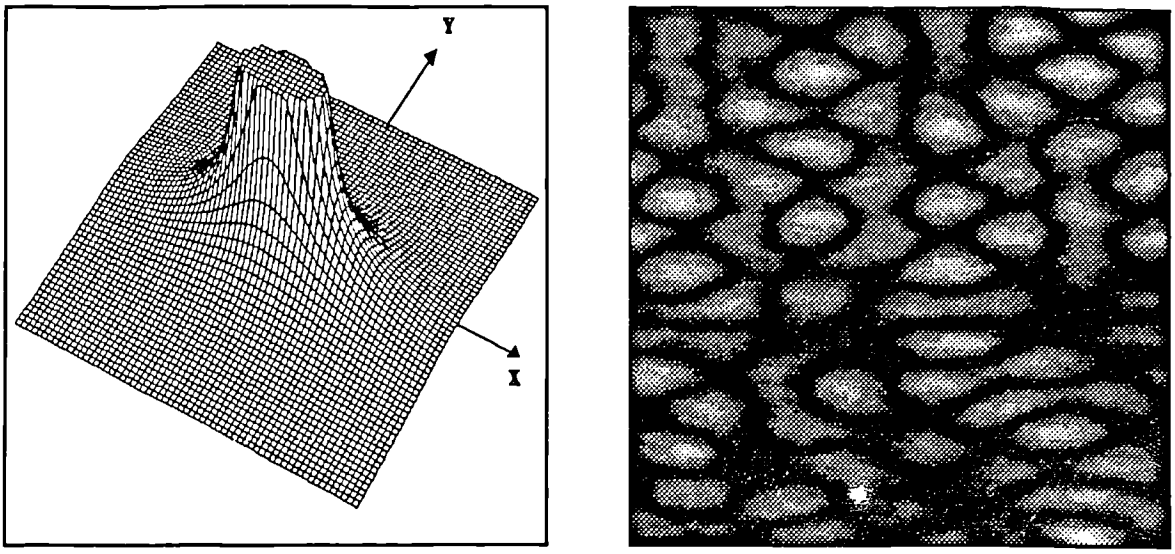


Figure 3.14. Model 5, (a) spectral plot(left), (b) texture image(right).

3.2.3. Shading and multifractals.

The basic fractal model (model 1) is severely limited in producing textures suitable for models of the seabed. However, it does have a use when combined with other textures to provide a model which can introduce shading effects into the texture. This is important for two reasons. First of all, in the present study of the seabed, data for a given region may arrive in two forms :-

a) Sonar or video image of the seabed.

b) Bathymetry (height) data of the seabed.

The bathymetry data is useful as an additional feature in helping to segment texture. For example, if the slope of an area of seabed is steep, certain textures will not be present, and corrections to the angle of the sonar (the slant range correction) can be made to produce images of the seabed where the topology has been taken into account. The ability to map the sonar onto the bathymetry data provides a useful aid to geologists and navigators, providing a visual image of the true seabed, and not requiring interpretation via charts or contour maps. Figure 3.15a shows a sonar image of the seabed, figure 3.15b is the bathymetry data plotted as a surface, and finally figure 3.15c shows these two images mapped together to produce the full image. In this final image, the sand like colour of the image has been added and the texture segmentation lines are also superimposed on the image.

This idea of texture mapping has been used by many authors to help produce more realistic images [3.6], [3.7]. The present use is novel in providing images of matched data sets where less interpretation is required to provide meaningful information.

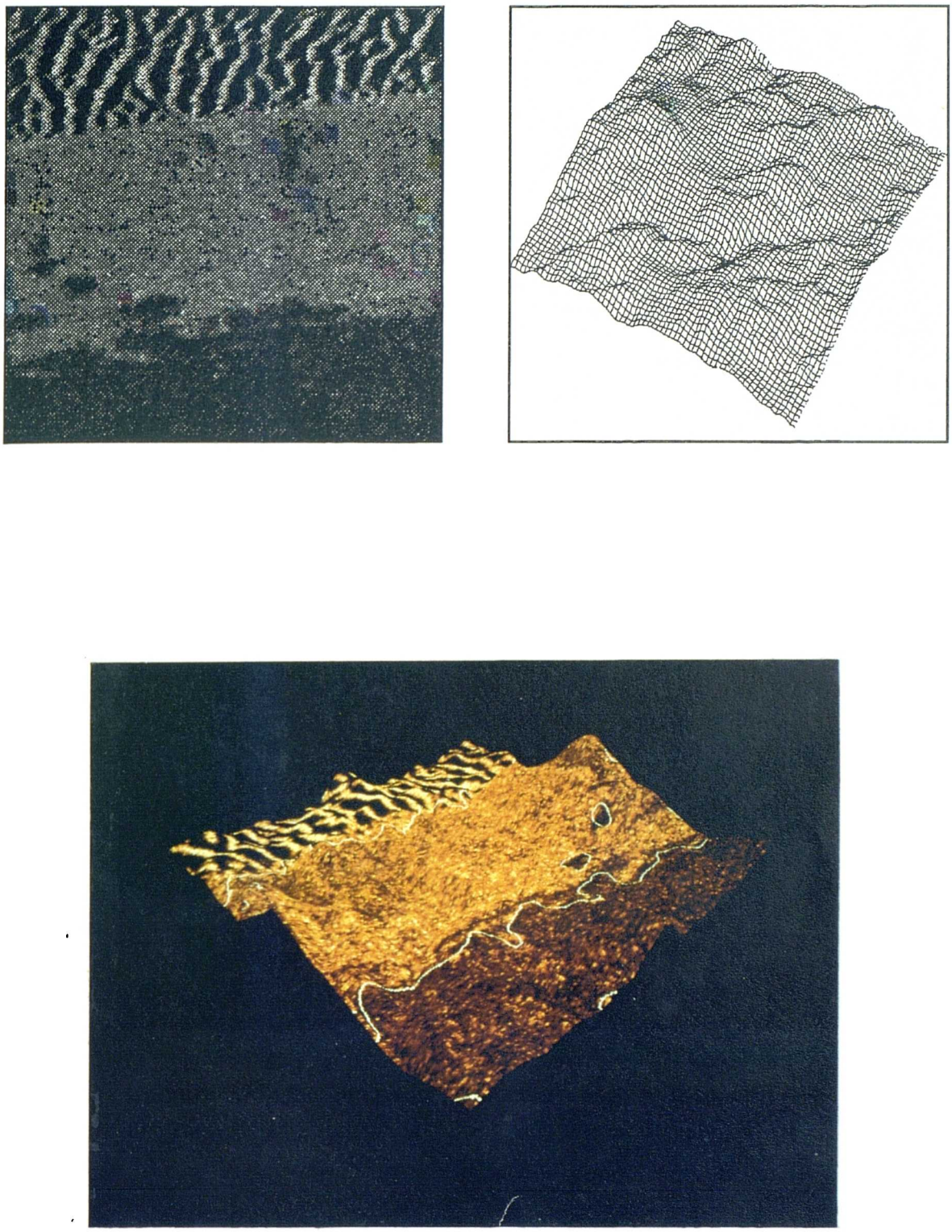


Figure 3.15a,b,c. Stages in producing a texture mapped image.

If the idea of texture mapping is extended to the synthetic images an extension to the modelling scheme emerges. In the image of figure 3.15c, the texture was in effect stretched over the surface of the height field. If, secondly, the height field is used to modulate the grey levels of an image this creates the underlying shading in the image while still preserving the original texture. In essence two fractal models have been combined to produce a multifractal model. This now increases the realism, provides a better test bed for the analysis schemes and produces a more flexible model. The effect of shading on the texture characteristics has not been intensively investigated. Kube and Pentland [3.8] have outlined the effect that shading can have on the fractal dimension of a fractal surface . This is relatively easily shown with this present model. It is difficult to predict the dimension of a model resulting from the addition of two fractals of differing fractal dimension, however, it is certain that the dimension of the original image is changed by the shading model. It may be more convenient to consider the combination of fractals as separate fractals. In the frequency domain the spectrum exists over a certain range of discrete values, and for fractals an analogous situation may be envisaged as the 'spectrum' existing over a certain range of fractal dimension. This work is concerned in the production of realistic images using the given tools, and the addition of the fractals to produce a more realistic image may be considered as combining the illumination and reflectance to produce an image. The shading model is equivalent to the illumination and the texture model is equivalent to the reflectance. To illustrate the ideas, an image is presented in figure 3.16. Figure 3.16a shows the grey level image and figure 3.16b shows the texture mapped surface image. Model 1 was used for the height or shading field and model 2 for the texture.

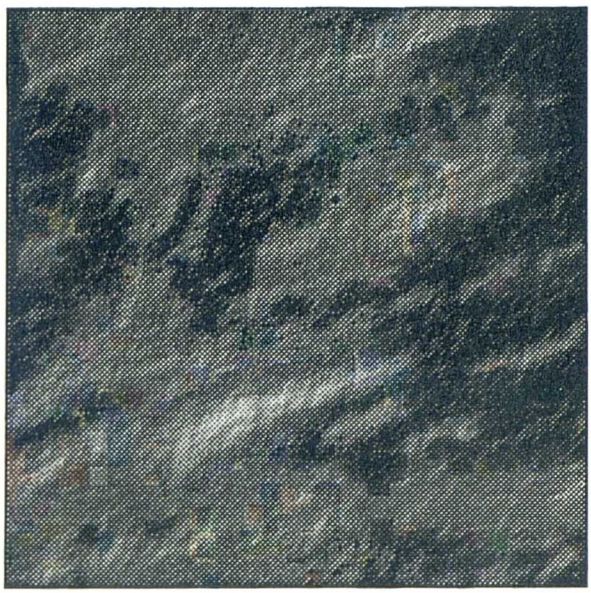


Figure 3.16a. Shaded texture image.

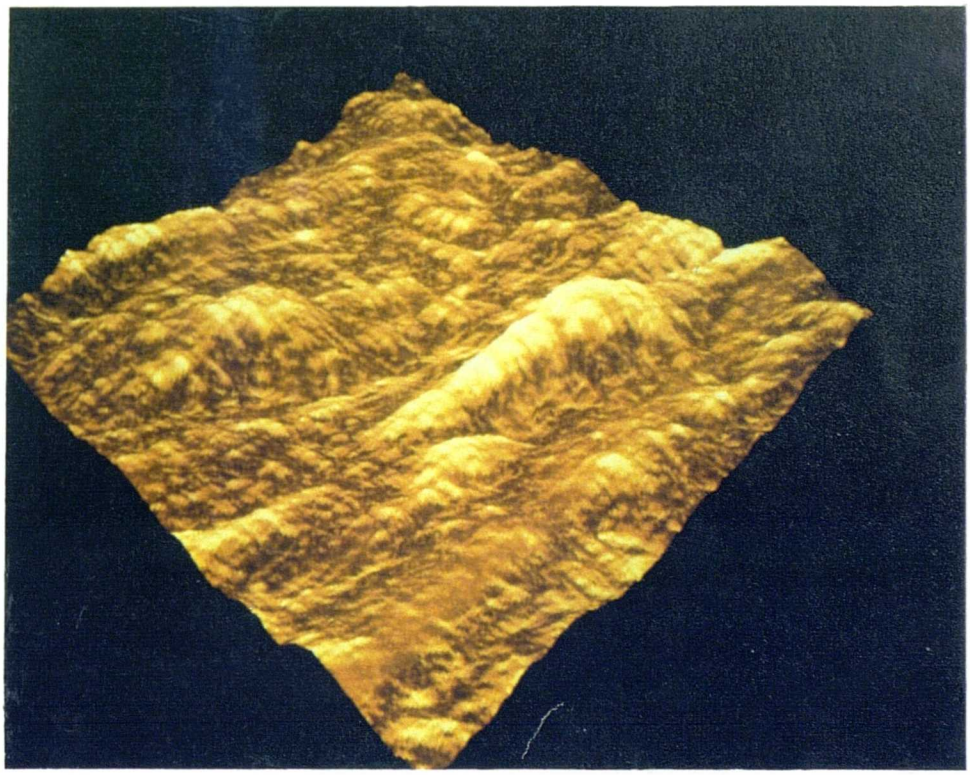


Figure 3.16b. Texture mapped surface of Figure 3.16a.

3.2.4. The production of multi-texture images.

Having achieved the synthesis of homogeneous textures and their shading for greater realism, consideration is given to their combination to produce a multi-textured image. Whilst it would be relatively easy to produce simple boundaries such as straight lines and circles, the creation of more natural looking boundaries is possible. In chapter 2 some 1D fractal lines of varying dimension were used and it was decided to use these again as the basis of the textural boundaries in the heterogeneous image. These boundaries may be relatively smooth (low fractal dimension) or very tortuous (high fractal dimension). The fractal dimension and the end points of the boundary can be controlled but its precise location on the image cannot be predicted. The algorithm adopted for creating these image boundaries was as follows :-

a) A fractal dimension for the boundary is chosen and a fractal line generated (see program *frac1d.c* in Appendix 3).

b) Start and end positions of the boundary in the image are chosen into which the textures are to be mapped.

c) Using rotation and translation, the line generated in (a) is transformed into the image between the points specified in (b).

d) (a), (b), (c) is repeated for other boundaries.

To convert a point with coordinates (x_1, y_1), to coordinates (x_2,y_2), by rotation through an angle(θ) and translation by (x_0, y_0) the standard formula given below is used

$$x_2 = x_1\cos\theta + y_1\sin\theta + x_0$$

$$y_2 = -x_1\sin\theta + y_1\cos\theta + y_0$$

Thus a point is taken from the fractal line and mapped into the image using the necessary rotation and translation. However, not all adjacent points on the boundary are continuous. To overcome this an algorithm (Bresenhams line drawing algorithm [3.9]) is used. This produces a straight line interpolation between two adjacent non-connected points which has the effect of producing a continuous boundary although slightly altering the original fractal line. Once the boundaries are produced, a texture may be mapped into one of the closed regions using a flood fill algorithm. These algorithms normally fill a region with a constant grey level, in this case however a corresponding grey level from the texture image is used. In this way a multi-textured fractal jigsaw can be produced. This makes excellent boundaries which provide a useful test bed for measuring segmentation accuracy. The image boundary may be further improved for visual presentation by transforming the grey level values of regions within a certain distance (number of pixels) of the actual boundary using weighted averages of the pixels in the neighbourhood. This produces a gradual transition from one texture region to another, creating a fuzzy boundary. Figure 3.17a shows an example of two boundaries mapped into a blank image producing three closed regions. Figure 3.17b shows those same regions filled with different textures. The fuzzy algorithm produces more natural boundaries for these textures. However it also distorts the original known boundary into one whose coordinates are not precisely known at every point. Figure 3.17b shows the boundaries without the application of the fuzzy algorithm.

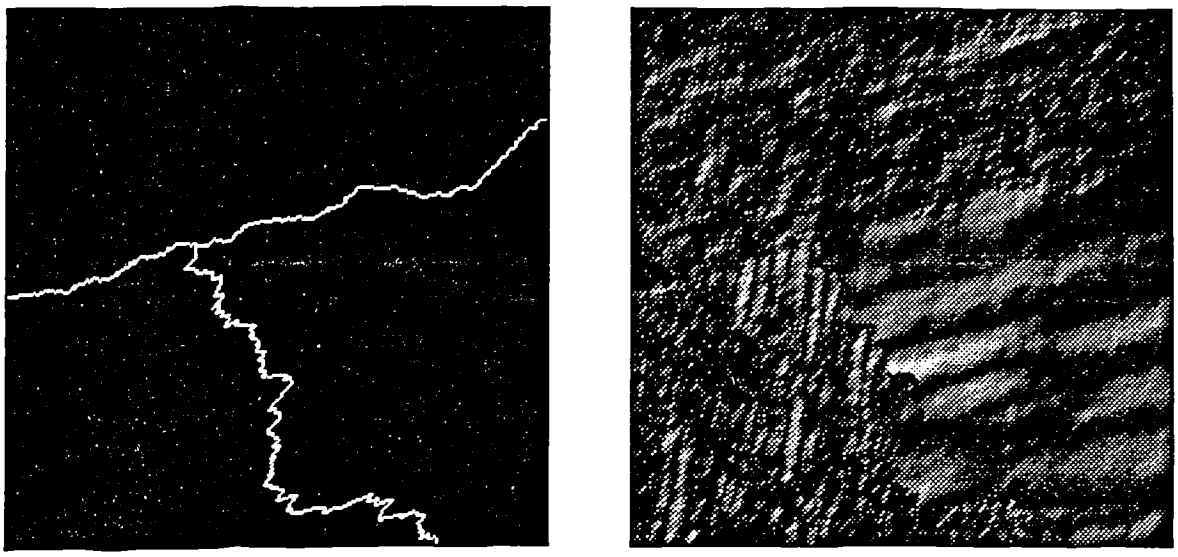


Figure 3.17. (a) Boundary image(left), (b) with textures(right).

3.3 The influence of phase on textures.

The last section used models that have a uniformly random phase distribution over the range $-\pi..\pi$. Further consideration is now given to the effects produced when the phase is not from such a distribution. Oppenheim and Lim [1.27] showed that in some circumstances the phase provides the most information, whilst the magnitude is of relatively small importance. For example, the two images shown in figures 3.18(a) and 3.18(b) are the normal original images. If these are transformed into their magnitude and phase components, their phases exchanged, and finally inverse transformed back to images, then the results shown in figures 3.18(c) and 3.18(d) are obtained. The effect is that of retaining the important structure of the image whose phase has been used.

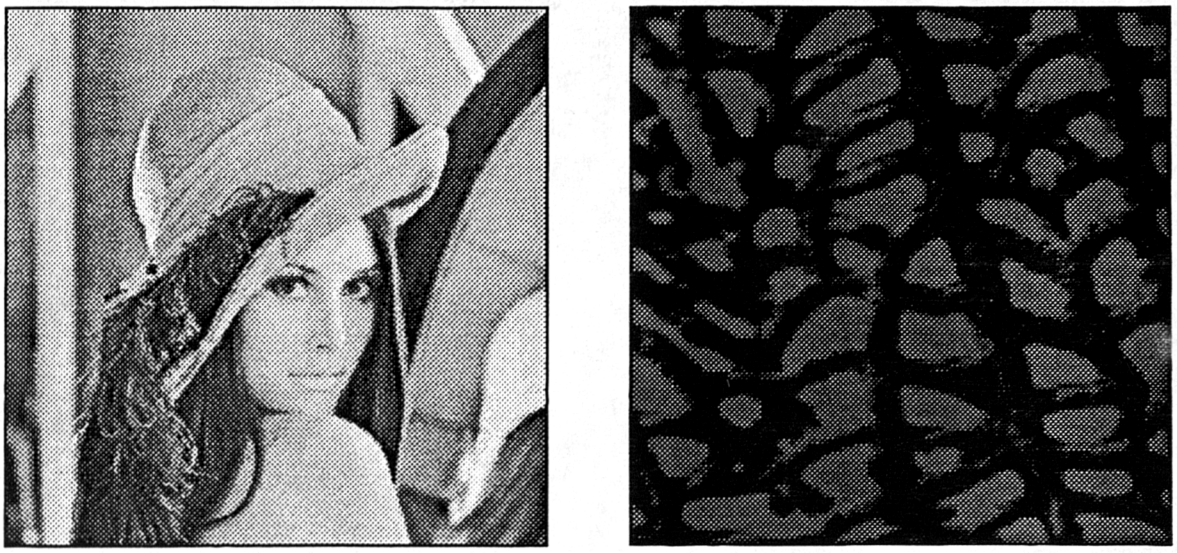


Figure 3.18a,b. Original images.

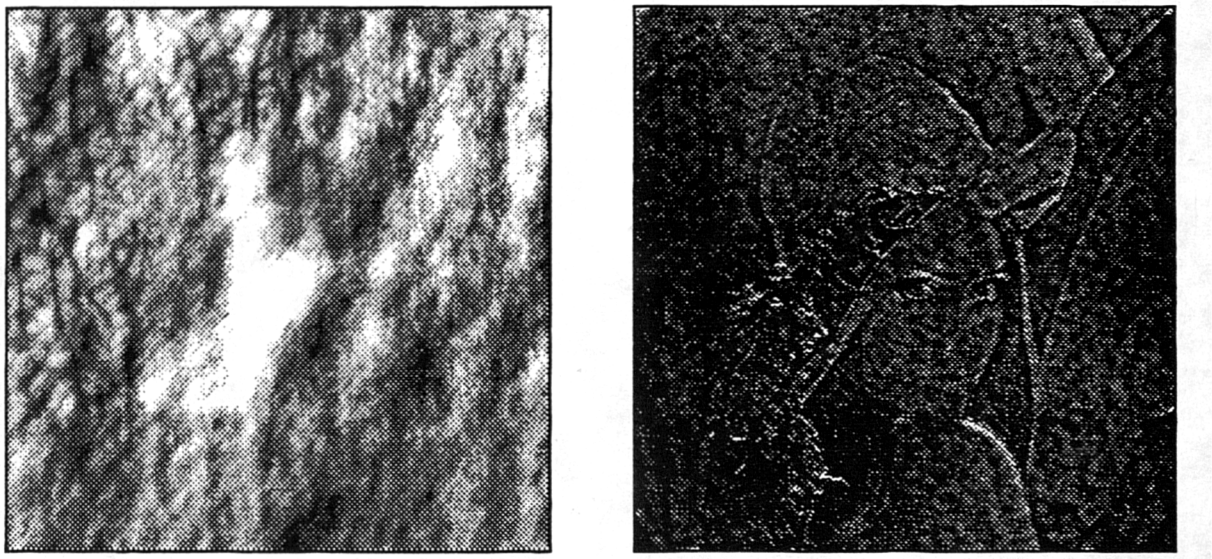


Figure 3.18c,d. Phase swapped images.

From this experiment, it appears that the phase is more important than the magnitude in retaining useful information. However, is this always the case ?

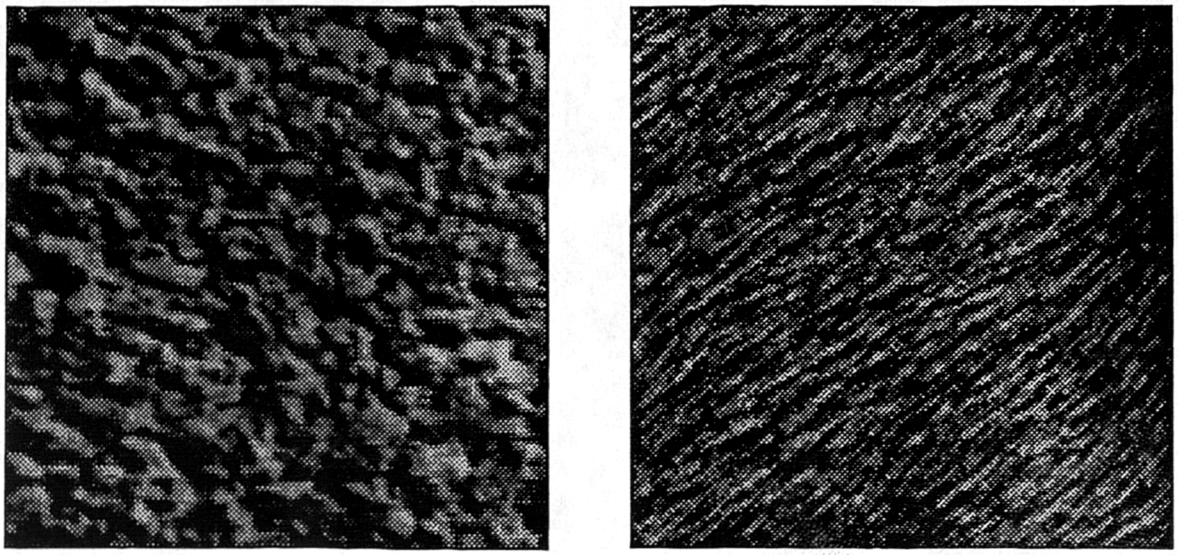


Figure 3.19a,b. Original images.

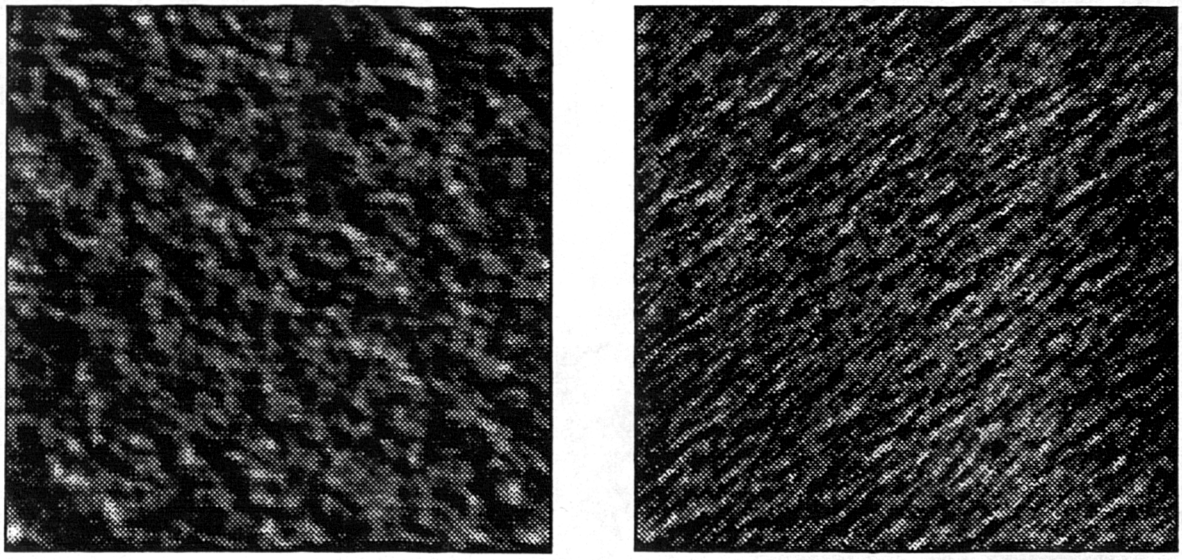


Figure 3.19c,d. Phase swapped images.

If the two textures shown in figure 3.19(a),(b) are examined for importance of phase, a different result emerges. Figures 3.19(c),(d) show the phase swapped images where figure 3.19(c) has the phase of figure 3.19(b), and figure 3.19(d) has the phase of 3.19(a). Clearly, in this situation, the phase has not produced the same effect as previously observed earlier with the Girl and Seaweed images of Figure 3.18. The effect may be explained by the amount of spatial structure of local features within an image. These features require positional information which is obtained from the phase spectrum. The lines or edges are the points where the Fourier components of the waveform are in phase with each other and the local energy is maximised. By randomising the phase at these points, the local structure is destroyed and the visual appearance of the image changes. To illustrate the above points, the signals presented in figure 3.20(a) and 3.20(d) are considered. In figure 3.20(a) the square wave signal has been modelled as the sum of a Fourier series where the phase between each of the components was constant. This may be expressed as follows :-

$$x(t) = a + A \sum_{k=1}^{n} \frac{1}{k} \sin\left(\frac{2\pi kt}{T} + \phi_k\right)$$

where a is the mean of the signal.

A is an amplitude factor.

k is an odd integer.

T is the period.

ϕ_k is the phase at frequency k.

n is the number of terms used in the series, in this case 50.

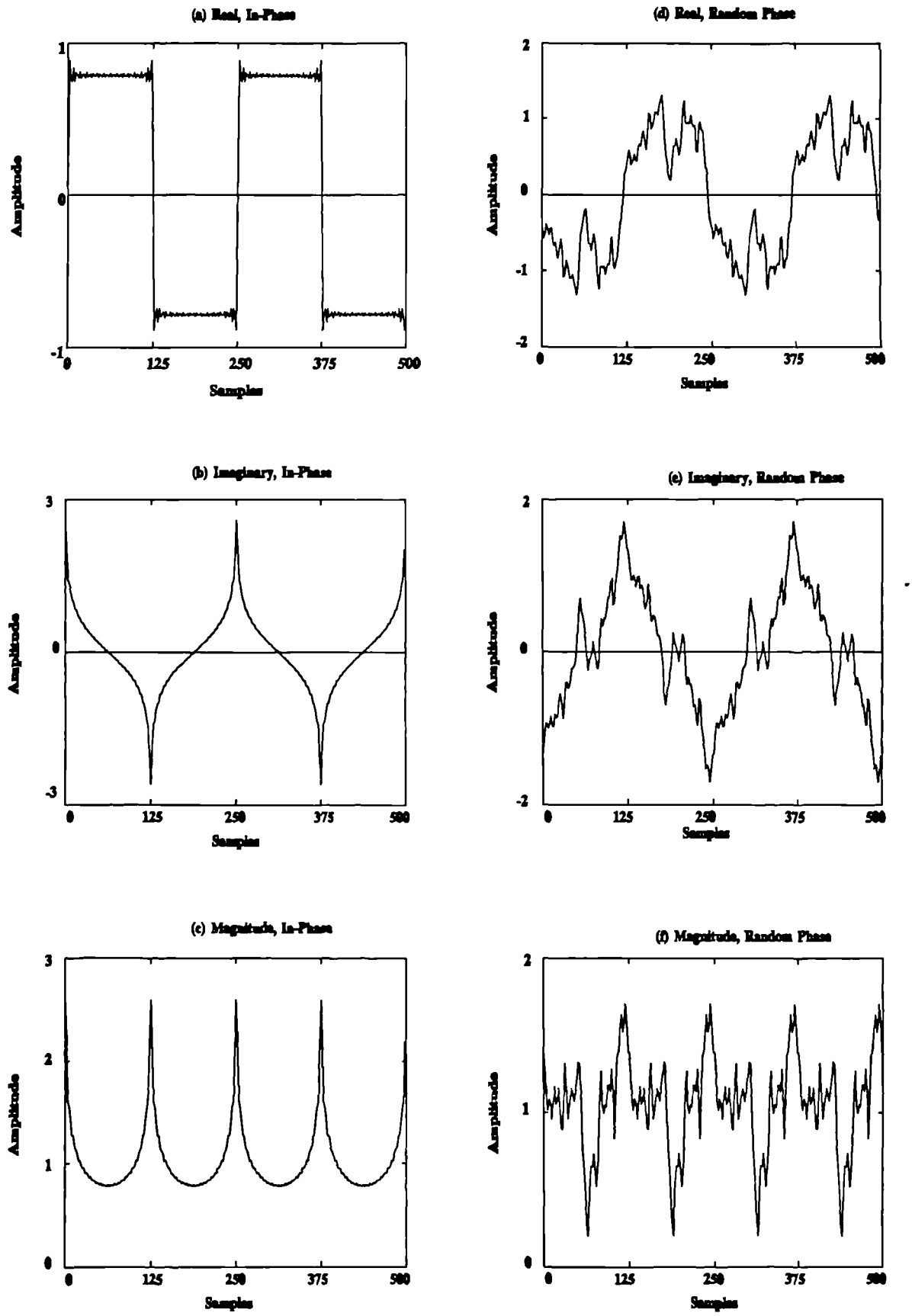


Figure 3.20 1D in-phase and random phase square wave.

For the square waveform, the phase (ϕ_k) was fixed at 0° for all frequencies, and two cycles of the waveform were produced. Since the phase was 0°, the edges of the square wave appear at integer multiples of π. Changing the value of ϕ would not change the square wave but merely alter its position as it is the phase spectrum that holds the positional information of an image. If the waveform is again generated, but for each frequency (k) the phase (ϕ_k) is chosen from a uniform random distribution with range -π..π, then the waveform of figure 3.20(d) is formed. The square waveform has been completely destroyed, the sharp edges at the integer multiples of π, and the positional information have disappeared. Thus in this case, the phase information is vital to the characteristics of the signal. As discussed previously the lines and edges are positions where local energy is maximised.. This may be shown by generating the quadrature form of the signal, ie the signal produced by phase shifting each of the Fourier components of the original signal by $\pi/2$. Mathematically, this phase shifted signal is the Hilbert transform of the original signal. Given a real signal $x(t)$ which may be considered as the real part of a complex signal $X(t)$, then

$$X(t) = x(t) + jx^*(t)$$

where $x^*(t)$ is the imaginary, or quadrature part of the signal. $X(t)$ is sometimes called the analytic signal. In the frequency domain, the situation is as follows:-

If $x(t)$ is real, then its Fourier transform may be represented by :-

$$x(t) = \int_{-\infty}^{\infty} B(\omega)e^{j\omega t}d\omega$$

$$= \int_{0}^{\infty} C(\omega)\cos(\omega t + \phi(\omega))d\omega$$

where $C(\omega) = 2|B(\omega)|$ and $\phi(\omega) = arg\ B(\omega)$, $\omega > 0$

therefore,

$$x^*t = \int_0^\infty C(\omega)\sin(\omega t + \phi(\omega))d\omega$$

$$\text{and} \quad X(t) = \int_0^\infty C(\omega)e^{j(\omega t + \phi(\omega))}d\omega$$

The amplitude spectrum of the complex trace, $C(\omega)$ vanishes for $\omega < 0$ and has twice the magnitude for $\omega > 0$, the phase $\phi(\omega)$ is unchanged. Thus given a real trace, the complex trace can be found by

a) Fourier transforming into the frequency domain

b) Setting the amplitude to zero for negative frequencies and doubling the amplitude for positive frequencies

c) Inverse Fourier transforming this modified frequency set

The resulting real part will be the original real signal, $x(t)$, while the imaginary part will be the quadrature signal ($x^*(t)$). The local energy at t can then be found from

$$X(t) = x(t) + jx^*(t) = A(t)e^{j\phi(t)}$$

$$magnitude(t) = \sqrt{x^2(t) + x^{*2}(t)} \qquad (3.2)$$

The corresponding phase relationship

$$\phi(t) = \tan^{-1}\left(\frac{x^*(t)}{x(t)}\right)$$

is often referred to as the instantaneous phase.

Figures 3.20(b) and 3.20(e) show the quadrature forms of the signals in 3.20(a) and 3.20(d) respectively, and figures 3.20(c) and 3.20(f) give the respective magnitude traces of the complex signals found using equation (3.2). From these, it can be seen that although the local energy is maximised at the edges of the square wave, this is not true for the random phase square wave.

The ability of the complex trace to detect changes in phase is extremely useful. For example, the sinusoid illustrated in figure 3.21 has a constant phase of 0° up to the first marked position. This phase is then linearly changed to 20° over the next 20 samples to the second marked position - a small rate of phase change. If the quadrature trace is found and then the magnitude of the resulting complex trace is obtained, the result shown above the sinusoid in figure 3.21 is obtained, showing clearly the region of the phase change. Much use of this technique has been made in seismic studies [3.10], [3.11] in the 1 dimensional case. This work has been extended by Granlund [1.26] and co-workers to 2D for use in image analysis.

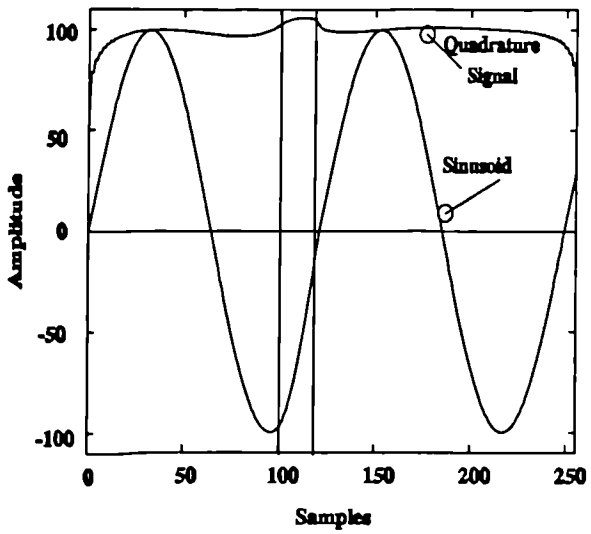


Figure 3.21 Detecting a phase change in a 1D signal.

The importance of the phase in textures can be further illustrated by observations on phase quantised images. If the phase is important for some textures, as was shown in figure 3.18, then changing the phase by quantisation should have a marked effect on the texture. Therefore the seaweed texture (which has a strong phase dependence) was subjected to some phase quantisation experiments. In the original image, the phase was quantised to 8 bits. In subsequent experiments, the number of bits used to represent the phase was reduced and the resulting effects observed. The results of these experiments are shown in figure 3.22 Figure 3.22(a) shows the original image and its 8-bit phase image. Figure 3.22(b) shows the corresponding set for 6-bit quantisation and figure 3.22(c) the results for 1-bit quantisation. It is only at 1-bit that the structure in the original image is destroyed. The remarkable result is that the general pattern of the phase images remains the same and the resulting spatial images retain their structure despite the phase being changed dramatically by restricting its values to fewer bits. In conclusion it is apparent that it is not the individual phases at each frequency which are important, but the relative phases at each frequency.

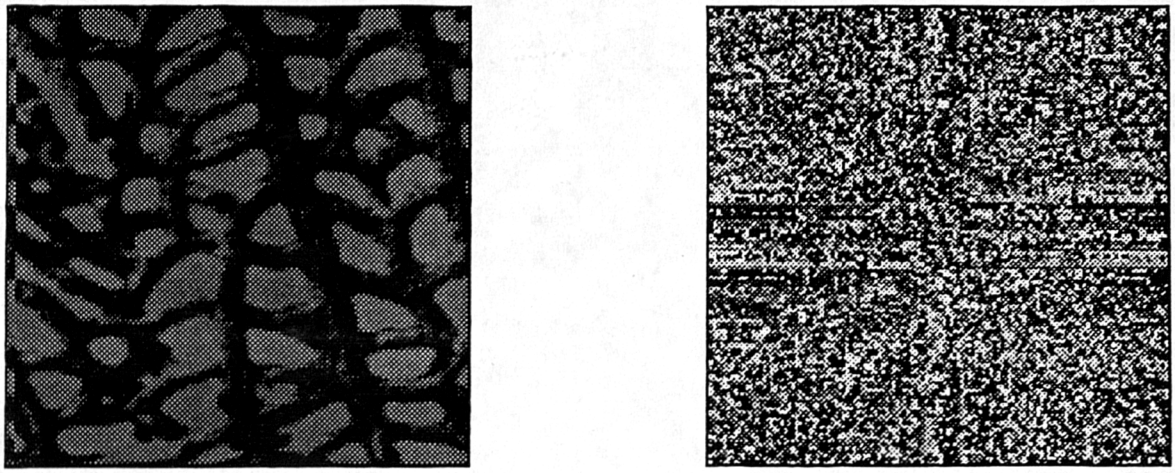


Figure3.22(a) 8-bit quantisation, image(left), phase(right).

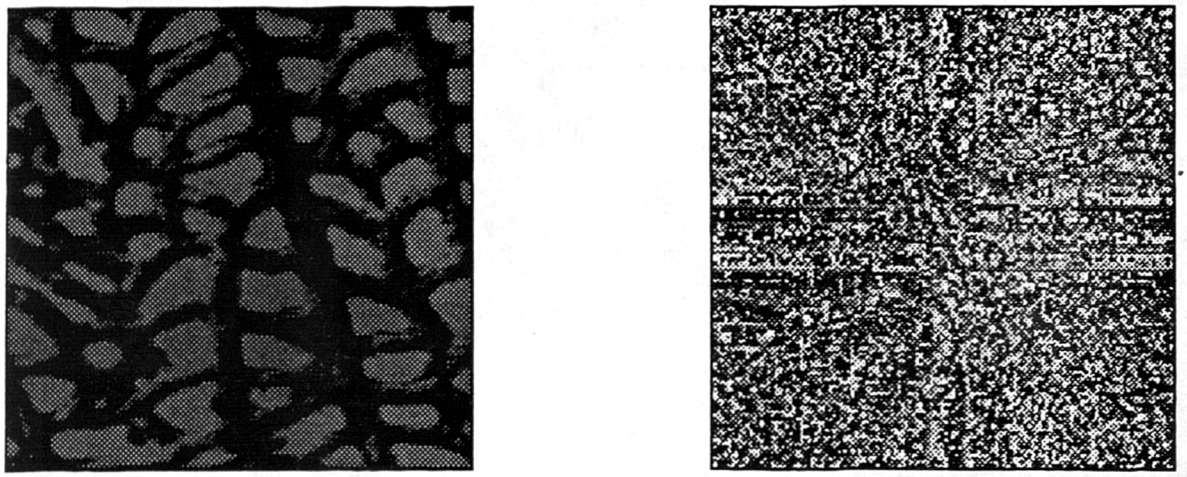


Figure 3.22(b) 6-bit quantisation, image(left), phase(right).

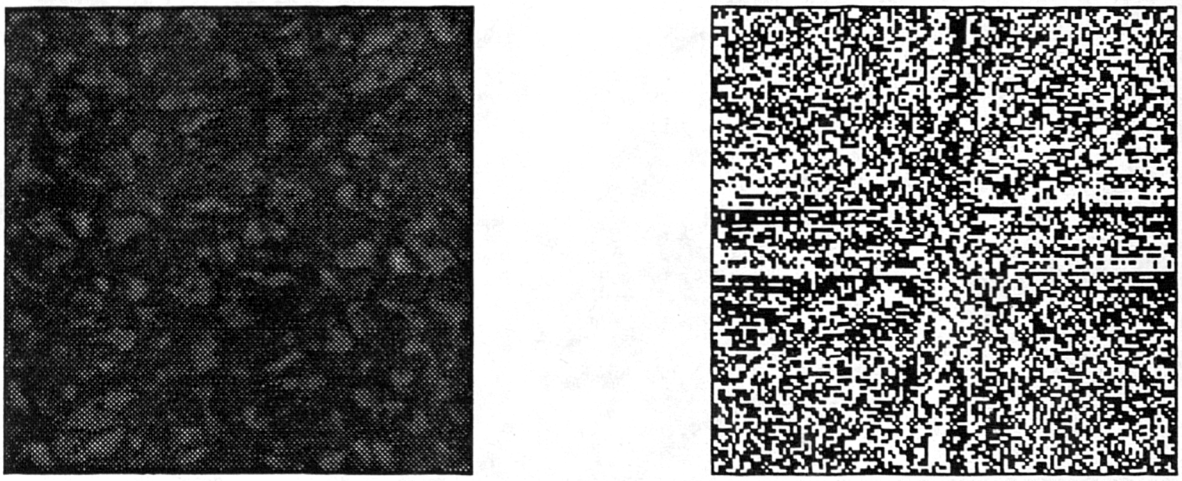


Figure 3.22(c) 1-bit quantisation, image(left), phase(right).

3.4. Extension and application of fractal ideas.

In extending the ideas of fractals, in line with multiresolution aspects of images, it was decided to examine two applications with regard to texture. One uses the spatial domain and the other the frequency domain.

Many other methods (apart from fractal) of modelling texture have been used by various authors. In particular Gagalowicz [1.2] has produced some excellent examples of synthesised textures based on measurements from the original texture. He used parameters obtained from the co-occurrence matrix [1.10]. In this 2D matrix, the size is determined by the range of pixel values in the image. Each element (i,j) of the matrix records the frequency of a pixel of grey level i being separated by a displacement dx from a pixel of value j. It follows that a matrix will exist for each value of displacement dx (directionality may be used also). If $f(x) = i$ and $f(x+dx) = j$ then the following equation is obtained -

$$E\left[|f(x+dx) - f(x)|\right] = \frac{1}{N}\sum_{i,j} |j-i|\, P(i,j|dx) \tag{3.3}$$

Where N is the total number of pixel pairs in the image sample and $P(i,j \mid dx)$ is the grey level co-occurrence matrix for displacement dx. E represents the expectation operator. Mandelbrot [1.40] has shown that a similar expression exists, namely

$$E\left[|f(x+dx) - f(x)|\right] . |dx|^{-H} = C \tag{3.4}$$

where C is a constant and H is a constant in the range 0 to 1.0. H is related to the fractal dimension D by $D = 3 - 2H$ (cf. equation (3.1)) in 2D. Thus by estimating the expectation expression in equation (3.3) for a given value of dx the value of H and hence D may be obtained from a log-log plot of E against dx. If this is done for a range of

synthetic fractal images of dimension 2.1 to 2.9 (in steps of 0.1) the graph shown in figure 3.23 is obtained, illustrating the approximation to the fractal dimension.

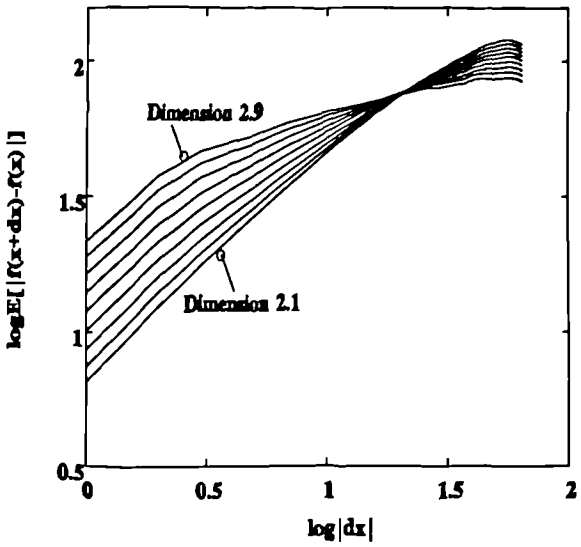


Figure 3.23. Estimation of fractal dimension by co-occurrence measurements.

The relevance of this is illustrated by the image shown in figure 3.24a. This is a typical sidescan sonar image taken off the South coast of England. Marked within it is a small square of size 32 * 32 pixels, (the original image is 512 * 512). The fractal dimension of this texture was estimated by the above technique and found to be 2.27. The texture had previously been segmented by the methods of chapters 2,4 and 5. Using this dimension and the variance of this estimate of D, a fractal interpolation in the spatial domain was used to obtain the images shown in figure 3.24b. The top row of this figure shows successively, the original 32 * 32 image, and expansions of X2, X4 and X8. Underneath the X8 image is the result of using a fractal dimension of 2 which corresponds to a linear interpolation. In the bottom left is a "blow-up" using pixel replication. This image shows that the fractal approximation has retained the underlying

textural structure and given a good approximation to expanding the original texture. Since equation (3.4) is an expectation operation, it has associated with it a distribution (assumed Gaussian for this aspect of the work) with a measurable variance. It is this variance that is used in the fractal interpolation. Using a method suggested by Mandelbrot, a point in 2D space may be interpolated using its neighbours and the value of H and σ obtained from equation (3.3) and (3.4). (σ^2 is the variance obtained from the measurements in equation (3.3)). The interpolation equation is given as -

$$f(i,j) = \frac{1}{4}\left[f(i-1,j-1) + f(i+1,j-1) + f(i-1,j+1) + f(i+1,j+1)\right] + \sqrt{1 - 2^{2H-2}} \cdot |dx|^H \cdot \sigma \cdot G$$

Where G is a Gaussian random variable with mean 0 and variance 1.

This aspect of the work perhaps has potential in object detection and boundary estimation.

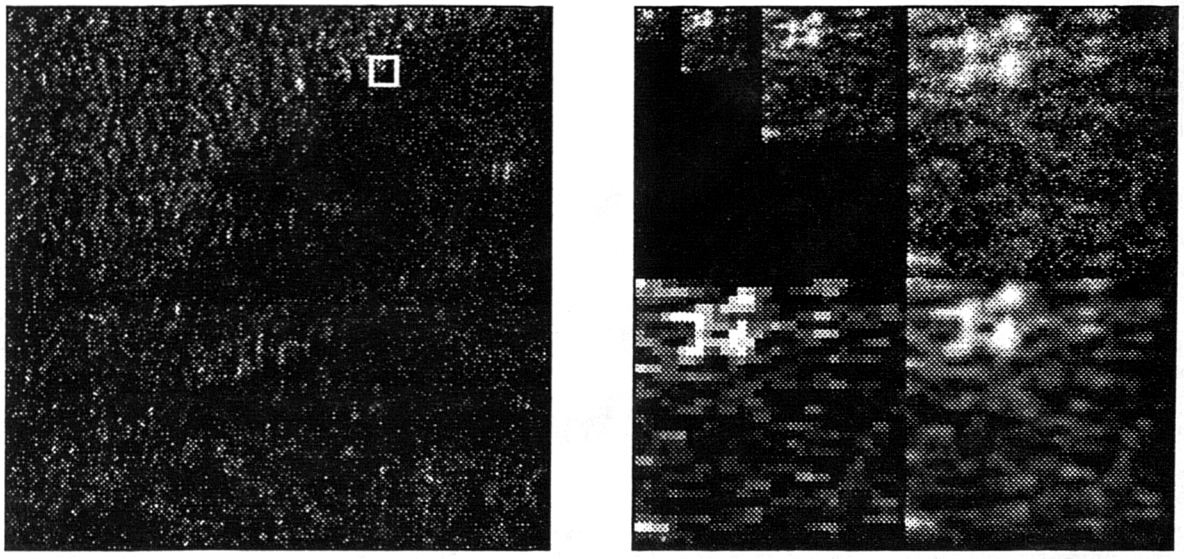


Figure 3.24(a) Original sidescan image. (b) fractal interpolation of region from (a).

Interpolation in the spatial domain thus allows areas to be enlarged while retaining the high frequency content, allowing a useful likeness to be obtained when a small region is zoomed. However, if the texture region was to be enlarged, but the scale was kept constant, then this would allow the texture to be extrapolated. To do this, consider interpolation in the frequency domain. As an illustration consider the original sidescan sonar image shown in figure 3.25a. This was segmented using the techniques of chapters 4 and 5 to give two distinct texture regions. A 32 * 32 pixel subsample of each region was taken and then interpolated in the frequency domain, just as had been done in the spatial domain. On inverse Fourier transformation of this interpolated region an image emerged that was twice as large but retaining the original texture scale. Using this technique and knowing the original classification a likeness to the original could be made as shown in figure 3.25b. This image was reconstructed from about 3Kbytes of data while the original was 256Kbytes, thus allowing a useful reduction for data storage. This aspect of texture regeneration is at present being examined more closely.

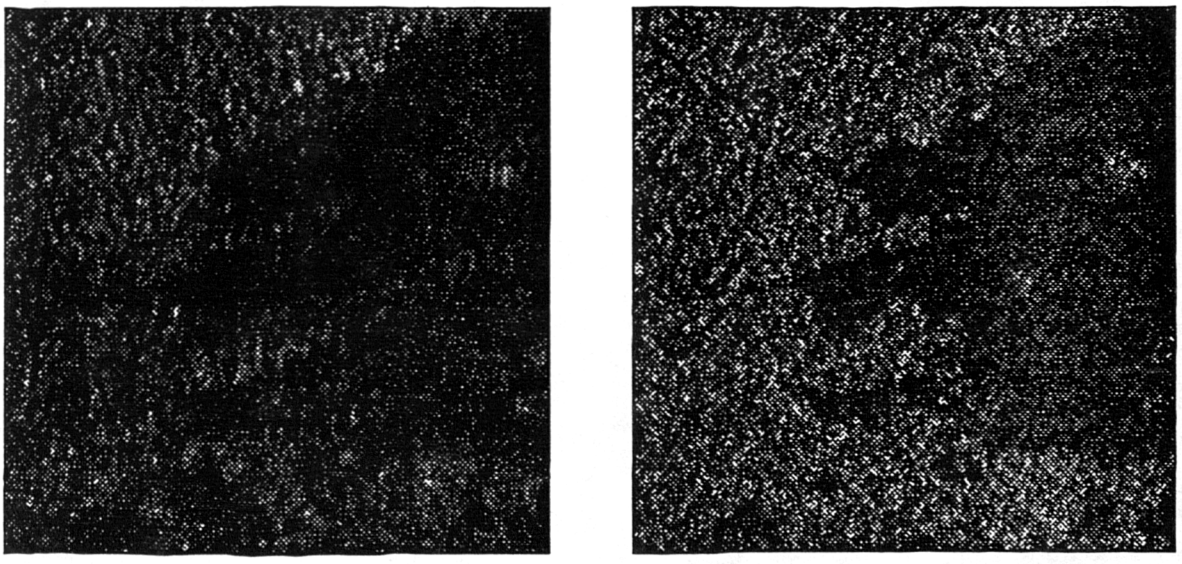


Figure 3.25 (a)Left, original image, (b)right, synthesised image.

3.5. Conclusions.

This chapter has presented some models for texture synthesis. These are based on fractal ideas. It was shown that the range of fractal dimensions is limited to narrow ranges for the production of natural looking textures. Directionality was shown to be important and was modelled by incorporating an anisotropy factor into some of the models. Phase was considered and many natural textures, including those from sonar records, may be modelled using a uniformly random phase distribution. However, it was also shown that many images cannot be modelled unless a more detailed knowledge of the phase is known. Finally, the use of frequency interpolation for texture reproduction, and frequency extrapolation, for texture magnification was shown, it being noted that frequency interpolation is equivalent to spatial extrapolation and vice-versa.

3.6. Authors' papers relevant to chapter 3.

1). L.M.Linnett, S.J.Clarke, D.N.Langhorne, C.Graham, *Remote sensing of the seabed using fractal techniques*, IEE, invited paper, for, Electronics Communication Engineering Journal, to be published.

2). L.M.Linnett, S.J.Clarke, *The use of fractal modelling in texture analysis*, IEE Colloquium, *The application of fractal techniques in image processing*, London, December 1990.

3). G.Shippey, L.M.Linnett, C.Graham, *Spectral attenuation measurements of high resolution seismic records using quadrature bandpass filters*, SEG 60th Annual International Conference, San Francisco, USA, 23-27 September, 1990, pp1075-1078.

CHAPTER 4

SUPERVISED PATTERN RECOGNITION

4.1. Introduction.

The assessment of a texture feature extraction algorithm must culminate in a technique in which the errors can be seen or calculated. This is necessary in order to evaluate the segmentation and the validity of the extracted features. The two most common techniques for feature assessment are supervised and unsupervised pattern recognition. This chapter will concentrate on the topic of supervised pattern recognition. A statement of the problem is as follows :-

An image has been transformed into one or more feature images and a pixel at any position has associated with it a feature vector $\mathbf{X} = \{ x_1, x_2, ..., x_n \}$, where there are n feature images. Within the original image are several textures. This information is used to assign one texture to the pixel under examination given its n dimensional feature vector. The methods used to achieve this are essentially statistical. Further techniques that are not strictly statistical may also be used, e.g. "nearest neighbour" matching of a pixel with its surrounding neighbours employing context.

The results from several techniques will be examined.

The key element of the supervised technique is the fact that information about the individual textures is known or obtained before a classification result is achieved. This is

usually done by the use of training areas within the original image. These are areas to which the observer has decided to assign specified textures. From these representative samples are derived the statistics of that texture, viz. the mean vector and the covariance matrix, for use in an appropriate model.

The most common classification scheme used in image processing is that of Maximum Likelihood Discriminant Analysis. A brief background to the technique is given below, and then a more detailed discussion of various aspects relevant to the image segmentation problem. Further references and introductions to Maximum Likelihood Discriminant Analysis may be found in [4.1], [4.2].

The basis for most discriminant models is the Gaussian distribution. The reason for this is based on the **Central Limit Theorem**, which states that if sufficiently many samples are successively drawn from any population the distribution of the sample values can be thought of, approximately, as an outcome from a normally distributed random variable.

4.2. Discriminant function theory.

The normal distribution for one variable is the familiar Gaussian distribution which has a density function given by

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where $p(x)$ is the probability density function of the random variable x. μ is the mean and σ is the standard deviation.

Extending this to the general n - dimensional multivariable normal density produces

$$p_i(x) = \frac{1}{(2\pi)^{n/2}|C_i|^{1/2}} e^{-1/2(X - \mu_i)^T C_i^{-1} (X - \mu_i)}$$

where

$p_i(x)$ = A measure of the probability for class i given the value x.

$\mathbf{X}$ = an n dimensional component column vector.

μ_i = the n component mean vector for class i.

T signifies transpose.

C_i = The $n * n$ symmetric positive definite covariance matrix for class i.

C_i^{-1} = The inverse of C_i.

$|C_i|$ = The determinant of C_i for class i.

Taking logarithms gives

$$\ln(p_i(x)) = -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln(|C_i|) - \frac{1}{2}(X - \mu_i)^T C_i^{-1} (X - \mu_i)$$

If the probability for a class i is known, i.e. p_i, then the above may be written as

$$\ln(p_i(x)) = \ln(p_i) - \frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln(|C_i|) - \frac{1}{2}(X - \mu_i)^T C_i^{-1} (X - \mu_i) \quad (4.1)$$

Usually the probabilities for each class are not known and are therefore assumed to be equal for each class. Thus the first term in equation (4.1) can be ignored. Similarly the second term is a constant and can also be ignored.

The general multivariate normal case for classes with different covariance matrices becomes :-

$$g_i(x) = -\frac{1}{2}\ln(|C_i|) - \frac{1}{2}(X - \mu_i)^T C_i^{-1} (X - \mu_i) \qquad (4.2)$$

where $g_i(x)$ is now the measure of probability for class i given the features x.

Expanding equation (4.2) gives,

$$g_i(x) = -\frac{1}{2}\ln(|C_i|) - \frac{1}{2}X^T C_i^{-1} X + X^T C_i^{-1}\mu_i - \frac{1}{2}\mu_i^T C_i^{-1}\mu_i \qquad (4.3)$$

This can be written as,

$$g_i(x) = X^T W_2 X + X^T W_1 + W_0$$

where

$$W_2 = -\frac{1}{2}C_i^{-1}$$

$$W_1 = C_i^{-1}\mu_i$$

$$W_0 = -\frac{1}{2}\ln(|C_i|) - \frac{1}{2}\mu_i^T C_i^{-1}\mu_i$$

i.e. a quadratic discriminant.

If the correlations between the variables are independent of the class, then the classes will have the same (or almost the same) covariance matrices. Equation (4.3) can thus be further simplified to give :-

$$g_i(x) = X^T C_i^{-1} \mu_i - \frac{1}{2} \mu_i^T C_i^{-1} \mu_i$$

$$= X^T W_1 + W_0$$

where

$$W_1 = C_i^{-1} \mu_i$$

$$W_0 = -\frac{1}{2} \mu_i C_i^{-1} \mu_i$$

i.e. a linear discriminant.

In this case the covariance matrix C is a pooled or average covariance matrix formed from the class covariance matrices as follows,

$$C = \frac{\sum_{i=1}^{k} n_i C_i}{(\sum_{i=1}^{k} n_i) - k}$$

Where there are k classes and n_i is the number of samples in class i.

In equation (4.2) the term $(X - \mu_i)^T . C_i^{-1} . (X - \mu_i)$ is often called the **Mahalanobis Distance** (D^2). This distance is a measure of the classification accuracy.

Note that if

$$D^2 = (X - \mu_i)^T C_i^{-1} (X - \mu_i)$$

then equation (4.2) can be expressed as

$$D^2 = -2g_i(x) - \ln(|C_i|)$$

This equation makes it computationally efficient to find the Mahalanobis distance once the value of the discriminant function has been evaluated. In practical terms, the greater the value of D^2 the more significant is the assignment of the pixel to that class.

In terms of the linear discriminant, the Mahalanobis distance represents the distance separating the 2 classes, i.e., the greater D^2, the greater the class separation and consequently the greater the probability of successful classification.

Let there be k pixels in the training area for class j and let there be n feature images. Thus each pixel **X** in that training area is composed of $x_1, x_2, \ldots x_n$

i.e.

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

The mean vector (μ_j) for class j is thus

$$\mu_j = \frac{1}{k}\sum_{i=1}^{k} X_{ij}$$

This mean vector gives the statistical centre of the class. To obtain the shape and size of the class in n - dimensional feature space, the covariance matrix C is required. This is calculated from the mean vector and the training set samples. The covariance matrix will be $n * n$ symmetric positive definite, and is calculated from the following equation.

$$C_j = \frac{1}{k-1}\sum_{i=1}^{k} (X_{ij} - \mu_j)(X_{ij} - \mu_j)^T$$

where T represents transpose and C_j is the variance-covariance matrix for class j.

Having obtained the mean and covariance measures for each class, the discriminant function for each class can be found using equation (4.3).

The next stage in the work is to classify each pixel using the feature images. Remember that each pixel **X** is in fact a vector [x_1, x_2, ... x_n] in equation (4.3). The value of each class discriminant function is evaluated for that pixel. The pixel is then

assigned to the class having the largest value discriminant i.e. the largest probability or maximum likelihood.

4.3. Factors affecting discrimination.

Having arrived at this stage, the method is apparently straightforward

a) Obtain one or more feature images.

b) Identify and gather the statistics of the training areas.

c) Apply the discriminant and obtain the classified image.

While this is an acceptable basis for the supervised classification technique, several questions are outstanding which remain to be answered in order to approach an optimum efficiency classification. Associated relevant topics which need to be considered are as follows :-

1) Is the Gaussian model appropriate for the data set ?

2) The effectiveness of the linear and quadratic discriminants.

3) Is there an optimum feature set ?

4) The size of the training areas.

5) The "don't know" situation.

6) The use of context.

We now consider each in turn.

4.3.1. Is the Gaussian model appropriate for the data set ?

This is a question which is rarely asked and appears to be even more rarely answered (at least in image processing). The great majority of workers assume that the

data is Gaussian and rely on the validity of the central limit theorem. The fact that alternative models are not often considered does not mean that model selection is unimportant; on the contrary, the correct choice of model may well determine the success or failure of the whole classification scheme. There is rarely a simple alternative to the Gaussian model. Few techniques, if any, within the field of image processing have been documented on alternatives to the Gaussian model, which is another reason for using this as a basis for a classification scheme. Finally there is, of course, the fact that the Gaussian model requires only two parameters, the mean and variance, and so can usually be implemented easily and efficiently. In trying to assess the present data, the features are first of all considered separately, i.e. the univariate case, where each feature is regarded as the marginal density of the multivariate case.

The features considered are those of one iteration of the directional fractal measure, detailed in chapter 2, on the image of the texture collage (figure 4.1), where the textures have been taken from Brodatz [1.35]. The directions are as previously outlined in chapter 2 (figure 2.18) and numbered 1 to 7, and these numbers are also the labels for the features. On the evidence of chapter 2, it is believed that use of directionality with the operator will produce a set of feature images which will allow segmentation when the features are presented to a statistical classifier.

The feature images themselves do not give an insight into the normality of the data, nor incidentally into the usefulness of that feature as a classifier. This can be seen from the two typical feature images shown (amplified threefold in intensity for illustration) in figure 4.2. These are for the collage image (figure 4.1) using directions 1 and 2 of the feature measures.

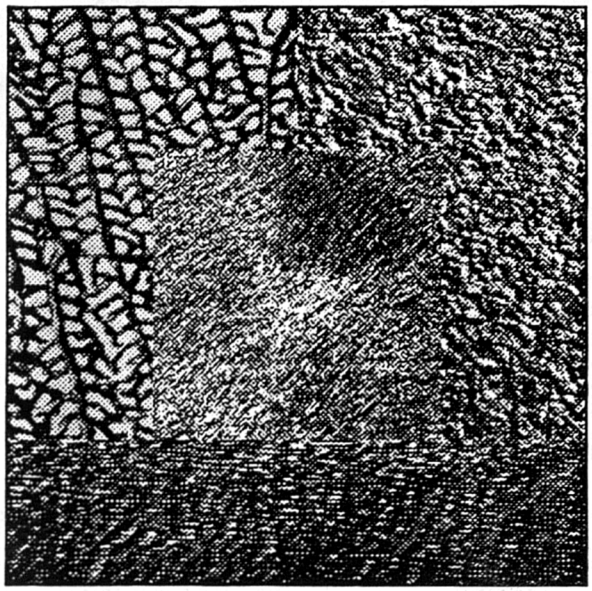


Figure 4.1. Texture collage image.

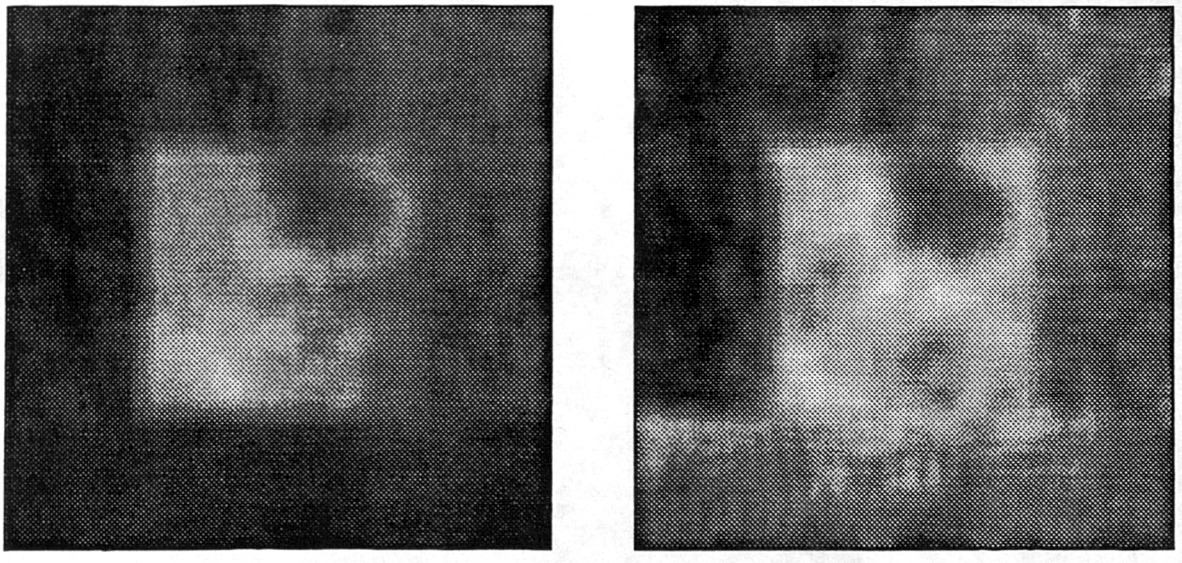


Figure 4.2. Two typical feature images from the texture collage image.

In the univariate case, there are several formal mathematical techniques which may be used to measure the significance of the fit of the data to a Gaussian model. These include the χ^2(chi-squared) and Kolmogorov-Smirnoff tests [4.3]. The technique most often adopted is a visual assessment of a histogram of the data, the rule being that if it looks Gaussian, then it is Gaussian! The eight histograms shown in figure 4.3 are representative of the four textures and features. Within each window are two histograms, one is the histogram of the feature data and the other the Gaussian histogram which has the same mean and standard deviation as the data.

As can be seen from figure 4.3, textures 1 and 2 appear to bear a good visual resemblance to their Gaussian counterparts while textures 3 and 4 are perhaps not quite so good but nevertheless probably visually adequate. In the χ^2 (chi squared) test, the values in each of the histogram bins are taken as the observed values (O). From the mean and standard deviation of these observed values, an equivalent Gaussian distribution is derived and the values in these bins are taken as the estimated values (E). χ^2 is a measure of the squared difference between the observed and estimated values, such that

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

where n is the number of bins in the histogram.

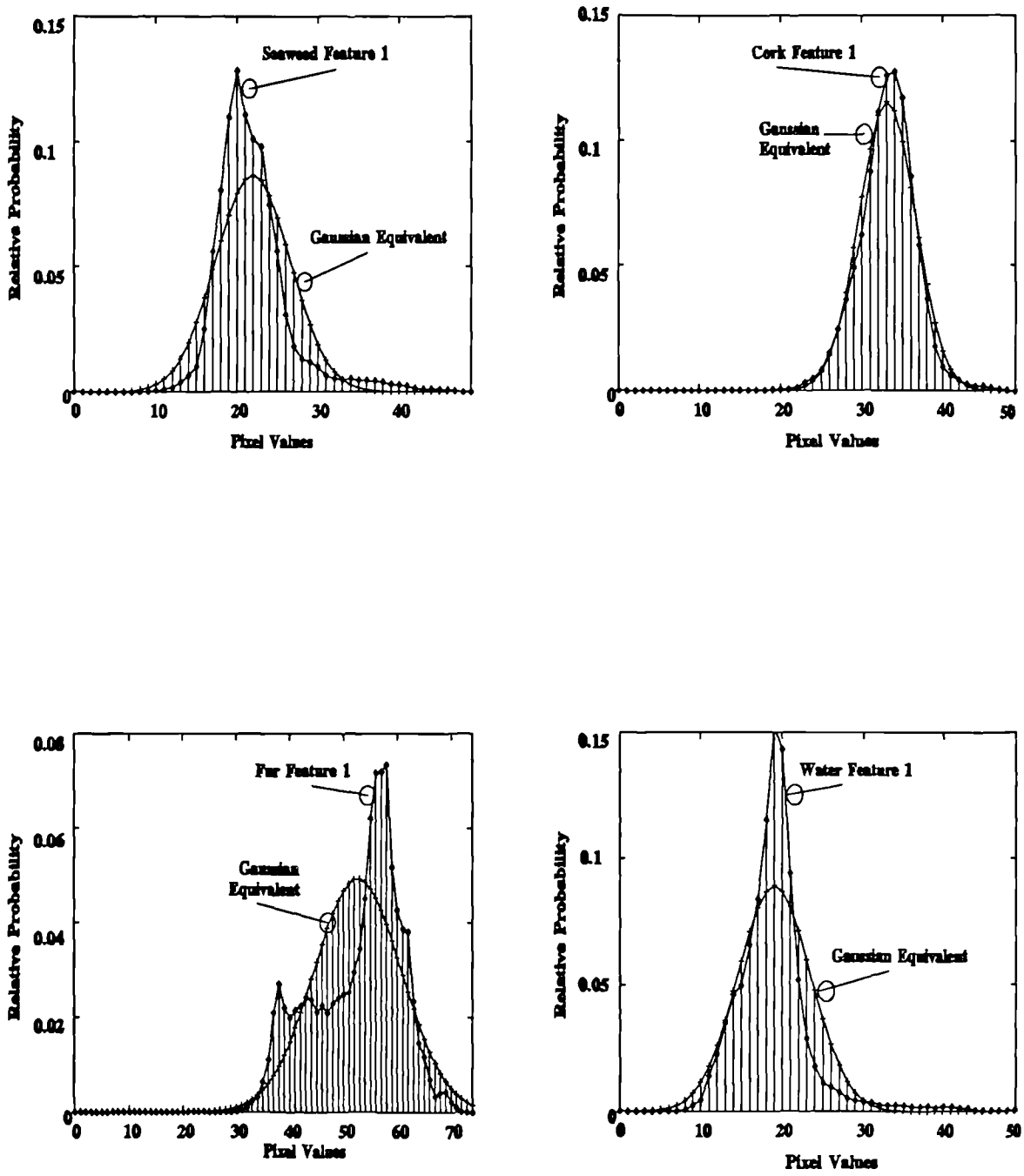


Figure 4.3. Selected feature histograms with their Gaussian equivalents.

To evaluate the significance of the fit, a number of "degrees of freedom " is required as well as χ^2. In the normal case this number is (n-3). The restriction of 3 in this case arises from the over-estimation of the problem by knowledge of the following 3 parameters.

1) the number of bins,

2) the mean,

3) the variance,

i.e. these 3 parameters are required to produce the estimated data. Table 4.1 shows the χ^2 values for each texture and each feature and the number of degrees of freedom for that feature. For completeness, the original image is included as feature 0.

When measured as a probability value, none was found to be significantly close to a Gaussian distribution. It is for this reason that the χ^2 values are given, even though the number of degrees of freedom vary. The probability function for χ^2 is $P(\chi^2 \mid \nu)$ and is the probability that the observed chi-square for a correct model should be less than a value χ^2. The probability function has limiting values $P(0 \mid \nu) = 0$ and $P(\infty \mid \nu) = 1$. The values are evaluated by using the incomplete gamma function defined as

$$P(a,x) \equiv \frac{\gamma(a,x)}{\Gamma(a)} \equiv \frac{1}{\Gamma(a)} \int_0^x e^{-t} t^{a-1} dt \qquad (a > 0) \qquad (4.4)$$

Here a represents $\chi^2/2$ and x represents $\nu/2$, where ν is the number of degrees of freedom. Convenient numerical methods for evaluating this are given in [4.3].

Texture	Feature	Degrees of Freedom	χ^2
Seaweed	0	143	42104
	1	36	8602
	2	39	1374
	3	40	6447
	4	45	2785
	5	45	10471
	6	51	3392
	7	57	7384
Cork	0	185	25031
	1	27	951
	2	33	2766
	3	41	1371
	4	35	1581
	5	32	1217
	6	35	2207
	7	38	1702
Fur	0	165	19497
	1	40	7323
	2	32	2165
	3	47	5540
	4	48	5938
	5	46	5997
	6	54	6611
	7	57	7343
Water	0	206	33051
	1	36	8071
	2	51	1823
	3	52	1996
	4	55	1444
	5	59	1322
	6	66	1076
	7	74	894

Table 4.1 χ^2 measures for each texture and feature.

It can be seen from this table that the features are nearer to Gaussian than the original image. However, when measured by the χ^2 test these features are far from Gaussian and this certainly casts doubt on the assumptions made from the visual assessment of the histogram and the use of a Gaussian model in the classification scheme. This analysis using single features merely acts as a pointer to the multivariate case. The multivariate case is certainly the most common and most practical technique, however the tests for multivariate normality are even less certain than those for the univariate case. For the univariate features to assist in the multivariate case, independence between all feature combinations would need to be proven and this is impractical if not impossible. While most univariate tests have evolved to produce multivariate counterparts e.g. '*t*' test to '*T*' test for the assessment of the difference of the population means, no single test has emerged for the testing of multivariate normality. Some tests have been put forward, for example Mardia [4.4] has looked at measures of multivariate skew and kurtosis and related these to a χ^2 distribution. More recently Smith and Jain [4.5] have put forward measurements of multivariate normality based on measures of the Euclidean distance between the sample points and a multivariate sample generated from the mean and covariance matrix of the sample.

4.3.2. The effectiveness of the Linear and Quadratic Discriminants.

The fact that the univariate features showed evidence of non-normality certainly indicates the need for caution when proceeding to the multivariate case and the use of the multivariate Gaussian in the classification scheme. At this point it was decided to evaluate the performance of the classifier based on the Gaussian model, since a good result would indicate a useful model. With this in mind, the Brodatz texture collage image (figure 4.1)

was selected for the test using the seven features previously mentioned. A training image was provided as shown in figure 4.4 and a ground truth image of the original was made as shown in figure 4.5.

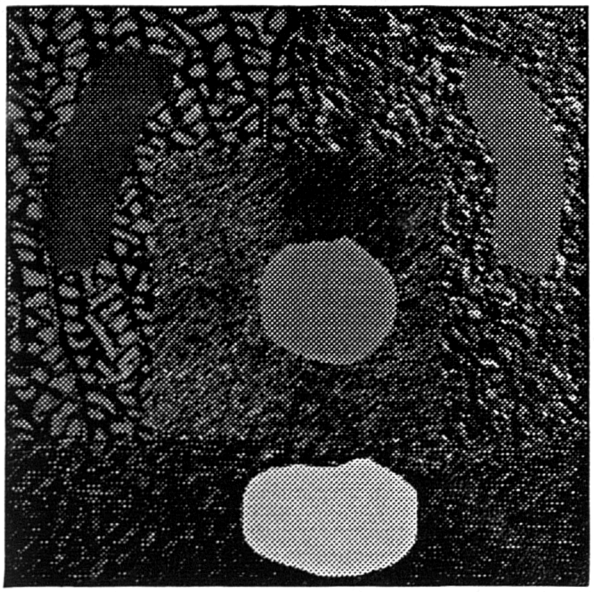


Figure 4.4. Training image regions.

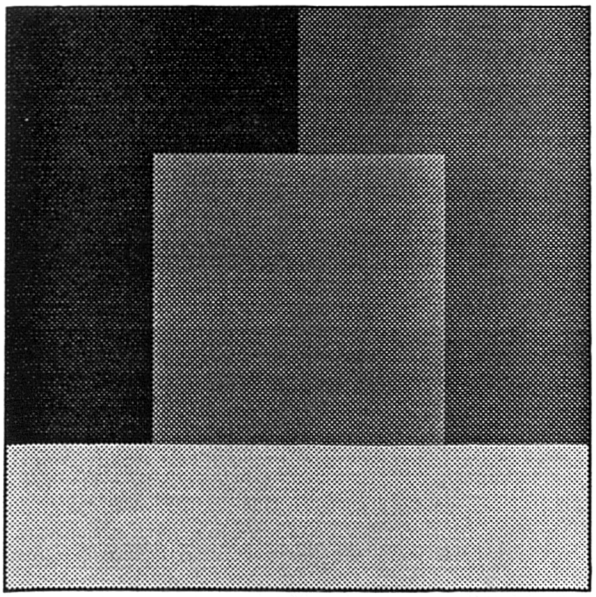


Figure 4.5. Ground truth image.

From the feature images and the training image, the feature statistics were obtained and both linear and quadratic discriminants were made. All possible feature sets were examined using first the linear and then the quadratic discriminant. The table showing all 127 possible results for the features is given in Appendix 1 table A1. The errors measured were based on the number of pixels that were mis-classified relative to the ground truth image. In terms of the Gaussian model, the best case results would seem to show that this is a reasonable approximation, indeed the size of the classification errors is excellent in comparison to that of similar classification schemes reported. The results relevant to this section are presented below.

		Errors (%)				
Discriminant	**Feature Set**	**Seaweed**	**Cork**	**Fur**	**Water**	**Average**
Linear	2,3,4,6,7	11.02	1.75	3.38	1.03	4.30
Quadratic	1,4,5,6	8.67	0.53	2.77	5.88	4.46

Table 4.2 Best overall feature sets for linear and quadratic discriminants.

Table 4.2 shows the best overall feature set measured in terms of the lowest average error over the four textures for both the linear and quadratic discriminants. The errors are given in percent and the numbers in the feature sets refer to the directions of the feature measures.

Table 4.3 shows the best individual feature sets and the errors (%) for each texture, again for the linear and quadratic cases. These would be the optimum if the features could be combined for a particular discriminant.

Discriminant	Texture	Features	Error(%)	Average(%)
Linear	Seaweed	1,2,3,5,7	6.12	2.44
	Cork	1,3,4,5,7	0.69	
	Fur	3,4,5,6,7	2.53	
	Water	1,2,4,5	0.41	
Quadratic	Seaweed	2,3,4,7	6.41	2.89
	Cork	1,4,5,6	0.53	
	Fur	1,2,3,7	0.04	
	Water	1,4,6,7	4.73	

Table 4.3 Best individual feature sets for the linear and quadratic discriminants.

It can also be seen that if discriminants could be combined in some way, then it might be possible to produce an error rate equal to that shown below. This particular point of combining the two models will not be pursued further here, however.

Discriminant	Texture	Features	Error(%)	Average(%)
Linear	Seaweed	1,2,3,5,7	6.12	1.78
Quadratic	Cork	1,4,5,6	0.53	
Quadratic	Fur	1,2,3,7	0.04	
Linear	Water	1,2,4,5	0.41	

Table 4.4 Optimum linear and quadratic sets.

The first observation is that there is little difference between the linear and quadratic discriminants, indeed, if anything the linear one is slightly better. This is important, as the linear discriminant executes at more than twice the speed of the quadratic discriminant. As it is a simpler model, it is more convenient for gaining an

understanding of the feature space. The fact that the linear case is perhaps better than the quadratic is on the face of it surprising, since it is intuitive that more terms should provide a better discriminant.

However, another explanation could be that the linear discriminant model is more robust than the quadratic in the multivariate case. As discussed in section 4.2, the linear case is merely a special case of the quadratic, where it is assumed that the covariance matrices for each of the features are the same (or at least from the same population). This equality of covariance matrices can be tested in the multivariate case. It does, however, assume multivariate normality. This test is important as it is also the key to many feature selection algorithms which also assume equal covariance and multivariate normality.

Assume that m variables are measured on each of k groups. For each group a covariance matrix C_i, $i = 1 \ldots k$, is computed. From these covariance matrices a pooled covariance matrix C_p is found, where

$$C_p = \sum_{i=1}^{k} \frac{(n_i - 1)C_i}{n - k}$$

where,

n_i = number of samples in group i.

$n = \sum_{i=1}^{k} n_i$ = total number of samples.

From this pooled covariance matrix a statistic (M) is computed which measures the difference of the logarithm of the pooled covariance and the average logarithms of the group covariance matrices. If the group covariances are similar, the difference is small and deviations from equality cause the statistic to increase. M is defined as follows

$$M = (n - k)\ln|C_p| - \sum_{i=1}^{k} (n_i - 1)\ln|C_i|$$

where $|C_p|$ and $|C_i|$ are the determinants of the associated matrices.

In order for this statistic to be used generally, a transformation to a χ^2 statistic is used [4.6], [4.7] where

$$\chi^2 = MQ$$

and Q is given as,

$$Q = 1 - \frac{2m^2 + 3m - 1}{6(m + 1)(k - 1)}\left[\left(\sum_{i=1}^{k} \frac{1}{n_i - 1}\right) - \frac{1}{n - k}\right]$$

The number of degrees of freedom(v) is given by,

$$v = \frac{1}{2}m(m + 1)(k - 1)$$

Applying this analysis to the present features produces the results shown in table 4.5. Only 7 of the possible 127 features sets are shown, the remainder producing similar results. In terms of percentage probability, all gave values much less than 0.1% when calculated using the gamma function previously described in equation (4.4). Thus the χ^2

values are quoted. The low probabilities indicate that the covariance matrices may not be considered equal and therefore a linear discriminant model should not be used.

Features	χ^2	Degrees of Freedom
1	32736	3
1,2	59218	9
1,2,3	156468	18
1,2,3,4	226369	30
1,2,3,4,5	241728	45
1,2,3,4,5,6	287146	63
1,2,3,4,5,6,7	315401	84

Table 4.5. χ^2 Values for equality of covariance matrices.

4.3.3. Feature selection.

Given a set of feature images, a decision has to be made whether to use all the features or to consider whether an optimum subset exists. Many workers have studied the feature selection problem [4.8], [4.9]. The problem must be considered for at least two reasons:

a) The computational effort rises rapidly as the number of features increases. If there are 'n' features then the number of possible feature sets is given by :-

$$\sum_{i=1}^{n} {}_nC_i = \sum_{i=1}^{n} \frac{n!}{i!(n-i)!}$$

Where ${}_nC_i$ represents the number of combinations of n objects taken i at a time.

Figure 4.6 shows the timings (in seconds on a microcomputer) for the execution of the linear and quadratic discriminants plotted against the number of features used. The image size was 512 * 512 pixels. As may be expected, each follows the order of

discriminant, the linear discriminant increasing linearly, the quadratic increasing as the square of the number of features. Thus either adoption of the linear discriminant as the model or reducing the number of features would be advantageous.

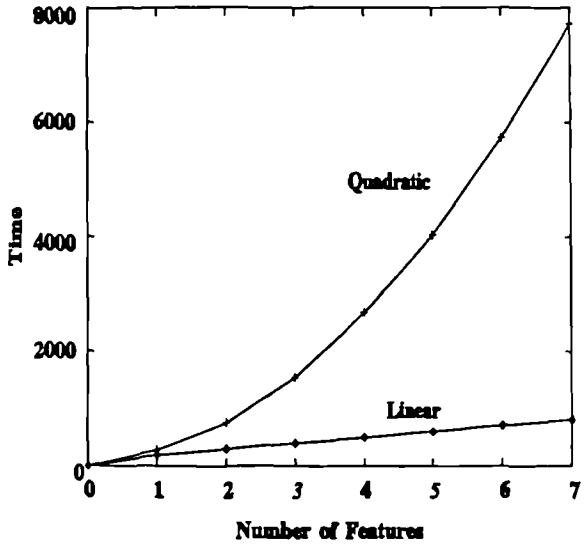


Figure 4.6. Relative timings of discriminants versus number of features.

b) As early workers in the field of pattern recognition discovered, increasing the number of variables does not necessarily increase the discriminating power of the classifier, so simply using all the features may not result in optimum performance.

As features are added, even though they add information, they may introduce errors into the assumed model. For example, if there is a multivariate Gaussian Model, a particular added feature may be non-Gaussian and give rise to serious errors in the classifier performance. Also the more complex the model becomes, the less the chance of understanding the need for particular variables and achieving a simpler model by, for example, a variable transformation. Since the purpose in this part of the work was to

study the feature measures, the seven directional measures were used as features with one iteration each i.e. seven features. With this number of features it was possible to study the exact relationship between features and to compare the results with those of some other search methods.

The results of this "exhaustive search" scheme are given in table A1 of appendix 1, where the numbers of the feature sets relate to the directions of the features. The graphs shown in figure 4.7, show the relationship between error rate and number of features. The results are shown separately for each texture from figure 4.1. Within each graph are two curves, one for the linear and one for the quadratic discriminant. The error rate given is the lowest for that number of features. While this result only applies to these features and this image, it does show the importance of choosing an optimal feature set.

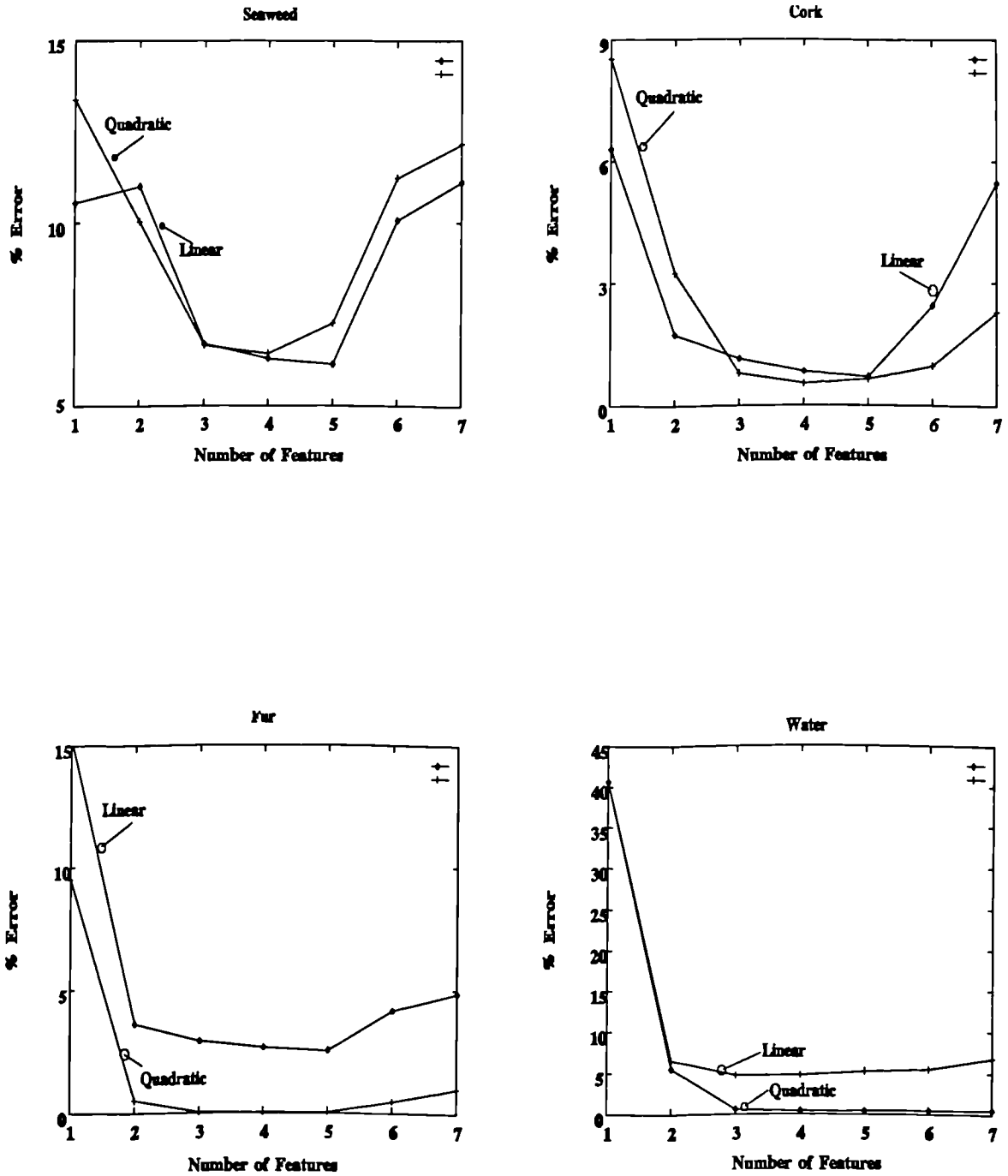


Figure 4.7. Optimal error rates versus number of features.

The "exhaustive search" method becomes impractical when the number of features is large (large in this context depends on the computer system, but for example on a micro-computer large would be about 5). Other methods must therefore be considered.

The optimum feature set search problem has been extensively studied [4.10], [4.11]. All the statistical procedures rely on multivariable normality and equality of covariance matrices. In all cases the measurement is for the separation of the groups. In the univariate case the test is for the separation of group means and is measured by the familiar t test [4.12]. With two variables, the counterpart is Hotellings T^2 test [4.12]. The extension to higher order multivariate problems is catered for with Wilks' lambda (Λ) test [4.12]. Essentially for a given set of variables, the separation between groups is measured. If the separation is significant then the variables are useful, if not, a new set of variables is considered or the offending 'bad' variables are identified and removed. There are two categories of feature searching:

a) Accelerated Searches.

These are based on the "branch and bound" method introduced by Narendra and Fukunaga [1.56]. These look at all possible subsets of the features, but do not explicitly evaluate all of them.

b) Stepwise Methods [4.13]

These exist as "forward" and "backward" methods. The "forward" method begins with a single variable and adds variables to produce a full set. The "backward" method begins with all the variables and eliminates a variable at each stage, measuring the separation of the groups at each stage.

One difficulty with both schemes is in deciding the termination criterion, i.e. the point at which the "best" set of features is found.

To illustrate the results of both methods, the present feature set was again used. Since the optimum was known a useful comparison could be made with at least these two standard feature selection algorithms.

(a) Stepwise Forward Search.

This method begins by selecting the best single variable. Several criteria for judging "best" are available, namely:-

a) Wilks' lambda (Λ).

This is a multivariate measure of group differences, it is defined as follows:-

$$\Lambda = \frac{|W|}{|W| + |B|}$$

where $|W|$ signifies the determinant of the pooled "within groups" sums of squares and cross products, and $|B|$ signifies the determinant of the pooled "between groups" sums of squares and cross products.

T is the total of sums of squares and cross products, -

$$T = W + B$$

The determinants effectively represent the volume that the data occupies in the sample space. Hence Wilks' lambda is the ratio of the volume within a group to the total volume of the data. Small values of Λ are more significant than larger values. Generally

the lambda value is converted to an F - value equivalent using a function first proposed by Rao [4.14]. It is defined as follows :-

$$Ra = \frac{1 - \Lambda^{1/s}}{\Lambda^{1/s}} \cdot \frac{1 + ts - n(G-1)/2}{n(G-1)}$$

where $t = m - 1 - (n+G)/2$

and

$$s = \sqrt{\frac{n^2(G-1)^2 - 4}{n^2 + (G-1)^2 - 5}}$$

$s = 1$ if $n^2 + (G\text{-}1)^2 = 5$

The number of degrees of freedom is $n(G\text{-}1)$ in the numerator and $1 + ts - n(G\text{-}1)/2$ in the denominator. m is the total number of samples, Λ is Wilks' lambda, n is the number of variables and G is the number of groups.

The statistic is exact for $n = 1$ or $n = 2$, or $G = 2$ or 3. An approximation to a χ^2 statistic can be made [4.15].

$$\chi^2 = -(m - 1 - (n+G)/2)\ln(\Lambda)$$

with $n(G\text{-}1)$ degrees of freedom.

b) Mahalanobis distance. (D^2).

This maximises the distance between the two closest groups, i.e. it maximises the minimum distances between each pair of groups. It is defined as :-

$$(\bar{X}_j - \bar{X}_k)^T.C^{-1}.(\bar{X}_j - \bar{X}_k)$$

Where $\bar{X}_j$, $\bar{X}_k$ are the means of the groups j,k respectively.

C^{-1} is the inverse of the pooled covariance matrix, and T signifies transpose.

Again this value is usually converted to an F - statistic with the following transformation :-

$$F = \frac{m_j m_k (m_j + m_k - n - 1)}{n(m_j + m_k)(m_j + m_k - 2)} \cdot D^2$$

which has n and $(m_j + m_k - n - 1)$ degrees of freedom in numerator and denominator respectively.

This conversion to an F statistic is useful since it takes into account the group sizes.

c) The Residual.

This is defined as :-

$$R = \sum \frac{1}{1 + D_{jk}/4}$$

where D_{jk} is the Mahalanobis distance between groups j,k.

Minimising R tends to separate groups which are close together. It was developed to take into account all the groups and is in effect a weighted average of all the D's between the groups.

d) The Conditional F ratio.

This measures how much a given variable contributes to the differences in group means, given the variable already included. The conditional F ratio is proportional to the ratio of the between groups sums of squares (B) and the within groups sums of squares (W). The larger the value the better the separation. The conditional F ratio is defined as :-

$$F = \frac{B}{W} \cdot \frac{m - g - q}{g - 1}$$

where m is the total number of samples, g the number of groups, and q the number of variables already selected.

As an example of the use of the stepwise forward procedure, the following shows an edited output from the program (see appendix 3, program *stepwise.c*), using the conditional F ratio as the measure of goodness. The seven features are as previously outlined.

Edited results of stepwise search program.

Step 1

Variables in equation = 0
Variables not in equation = 7
Feature 1 F-Ratio = 3.35980e+05 df=3, 79300
Feature 2 F-Ratio = 8.11114e+04 df=3, 79300
Feature 3 F-Ratio = 6.87969e+04 df=3, 79300
Feature 4 F-Ratio = 8.17254e+04 df=3, 79300
Feature 5 F-Ratio = 1.10046e+05 df=3, 79300
Feature 6 F-Ratio = 8.56583e+04 df=3, 79300
Feature 7 F-Ratio = 1.13510e+05 df=3, 79300

Variable 1 has largest F-Ratio 3.35980e+05
Discriminant functions

class	1	2	3	4
	4.16336e+02	6.81354e+02	1.08887e+03	3.64084e+02
const =	-1.68750e+01	-3.71135e+01	-1.76805e+02	-1.20021e+01

Step 2

Variables in equation = 1
Feature 1 F-Ratio = 3.35972e+05 df=3, 79298

Variables not in equation = 6
Feature 2 F-Ratio = 9.76112e+04 df=3, 79299
Feature 3 F-Ratio = 5.37306e+04 df=3, 79299
Feature 4 F-Ratio = 5.69468e+04 df=3, 79299
Feature 5 F-Ratio = 6.40705e+04 df=3, 79299
Feature 6 F-Ratio = 4.75290e+04 df=3, 79299
Feature 7 F-Ratio = 5.80456e+04 df=3, 79299

Variable 2 has largest F-Ratio 9.76112e+04
Discriminant functions

class	1	2	3	4
	2.89789e+02	4.03910e+02	1.10838e+03	8.48603e+01
	1.48937e+02	3.26535e+02	-2.29611e+01	3.28630e+02
const =	-2.22545e+01	-5.72291e+01	-1.76270e+02	-3.33069e+01

Step 3

Variables in equation = 2
Feature 1 F-Ratio = 3.91572e+05 df=3, 79297
Feature 2 F-Ratio = 9.76088e+04 df=3, 79297

Variables not in equation = 5
Feature 3 F-Ratio = 1.13595e+05 df=3, 79298
Feature 4 F-Ratio = 9.89444e+04 df=3, 79298
Feature 5 F-Ratio = 1.71906e+05 df=3, 79298
Feature 6 F-Ratio = 4.76699e+04 df=3, 79298
Feature 7 F-Ratio = 9.53834e+04 df=3, 79298

Variable 5 has largest F-Ratio 1.71906e+05
Discriminant functions

class	1	2	3	4
	5.93844e+02	1.07175e+03	1.21813e+03	7.42303e+01
	4.64394e+02	1.01943e+03	9.09096e+01	3.17601e+02
	-4.04624e+02	-8.88742e+02	-1.46057e+02	1.41460e+01
const =	-4.02818e+01	-1.18098e+02	-1.91658e+02	-3.27142e+01

In comparison the exhaustive search based on actual classification gives the results shown in table 4.6.

Number of Features	Exhaustive	Stepwise
1	1 (25.85)	1 (25.85)
2	1,5 (10.25)	1,2 (11.88)
3	1,5,2 (6.13)	1,2,5 (6.13)
4	3,4,5,7 (4.89)	1,2,5,6 (5.57)
5	2,3,4,6,7 (4.30)	1,2,5,6,7 (5.64)
6	1,2,3,4,6,7 (4.69)	1,2,5,6,7,3 (5.65)
7	1,2,3,4,5,6,7 (5.50)	1,2,5,6,7,3,4 (5.50)

Table 4.6. Comparison of stepwise and exhaustive search methods.

The figures in brackets are the percentage error rates using that set of variables and a linear discriminant, the model upon which the stepwise method is based.

It is noted that while it is only in agreement with the 1 and 3 feature sets (ignoring the common 7 set), the error differences are small and may be considered acceptable.

(b) The branch and bound method.

In this technique a tree structure is built with all the variables at the top (or root node). Wilks' lambda is usually used as a measure of significance. This is calculated for all the variables for all the groups for the root node. The next level down has one fewer variable and so on down to the required number of variables. The search begins by computing the Λ values for all the k variable combinations at the bottom of one of the main branches, k being the number of variables required in the final subset. The minimum Λ is noted (the lower the value of lambda the more significant the result). The search

continues by moving up a level on this branch calculating the Λ value for feature sets at this level on this branch, eventually ending at the root node. Another main branch is started and Λ computed for levels moving down this branch. When a Λ is found which is greater than the minimum for the *k* set case, the search on that branch is terminated and another branch is searched. In this way not all combinations of variables need to be examined, since implicit in the algorithm is the fact that given a Λ at a certain level on a certain branch, then the value will only increase (i.e. become less significant) as one moves down the branch. Implementing this scheme for the present data set produced the results shown in table 4.7, again shown in comparison to the exhaustive search method.

Number of Features	Exhaustive Search	Branch and Bound
1	1 (25.85)	1 (25.85)
2	1,5 (10.25)	1,2 (11.88)
3	1,2,5 (6.13)	1,2,5 (6.13)
4	3,4,5,7 (4.89)	1,2,5,6 (5.57)
5	2,3,4,6,7 (4.30)	1,2,5,6,7 (5.64)
6	1,2,3,4,6,7 (4.69)	1,3,4,5,6,7 (4.88)
7	1,2,3,4,5,6,7 (5.50)	1,2,3,4,5,6,7 (5.50)

Table 4.7. Comparison of branch and bound and exhaustive search methods.

This differs from the stepwise method results in only the 6 feature case, and agrees with the exhaustive search scheme in only the 1 and 3 feature set cases. Again it should be emphasised that the exhaustive scheme is the classification result. Given an exhaustive search based on Wilks' lambda, then the results would have agreed with the branch and bound method.

At this point the performance of the feature selection criteria may be evaluated. The performance is apparently not optimum when compared with the results of the exhaustive search, agreement occurring with only 2 feature sets. However, the error in the search criteria is not too bad. To illustrate this, the results are presented graphically in figure 4.8. Several observations can be made from this graph.

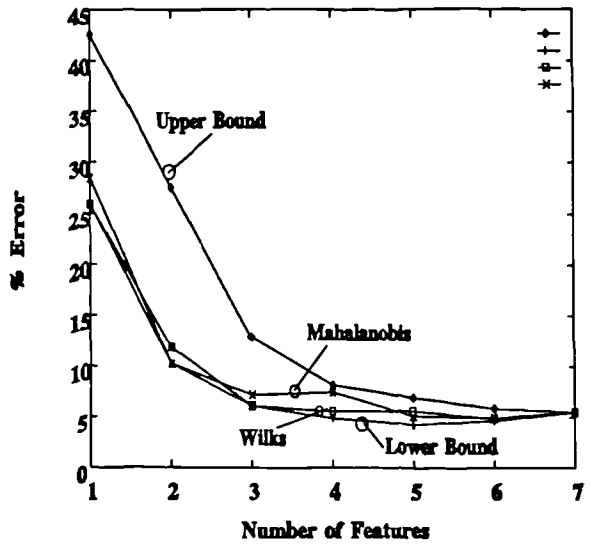


Figure 4.8. Overall error rate versus number of features.

First of all the lower and upper bounds of the classification result can be seen. The lower bound shows a minimum at 5 features and thus shows that adding more features does not necessarily improve the classification, indicating the need for feature selection. The upper bound however shows no minimum and the range of errors between the lower and upper bound is decreasing towards the common feature set of 7. The Wilks' curve tends to remain closer to the minimum error curve and the average displacement from it is only 0.55% in absolute terms, so that while it did not agree exactly with the minimum

error feature selection, the errors incurred with the Wilks' criterion were minimal. Again it should be stressed that these differences are almost certainly due to the assumption of a Gaussian model with equal covariance matrices. Since the minimum error curve is also based on this criterion, the comparison is valid.

The curve using the Mahalanobis distance is also shown, and while it differs from the Wilks' curve, it also lies close to the minimum error curve. Its average displacement from the minimum error curve is 1.04% in absolute terms. Using the stepwise method with the conditional F ratio as the measure of significance produced an average displacement of 0.42% from the minimum error curve.

In summarising the feature selection problem several points emerge. First of all since the methods all rely on the Gaussian model and equality of group covariances, the data sets should be examined for gross departures from normality. Such departures will almost certainly produce a non-optimum result. Even with these departures from normality, reasonable estimates can be obtained using the stepwise forward and branch and bound methods. Whilst there is little difference (at least in this example) in accuracy between the methods and indeed in the criteria used by the methods, the stepwise method is preferred since it is quicker than the branch and bound method and much more easily implemented.

4.3.4. The size of the training area.

This topic is important since a training area is usually chosen on an *ad hoc* basis. It was decided to check what effect (if any) the training area had on the resulting classification and to see if the effect of the choice of the training area could be minimised.

In statistical terms the training areas are required to form the population estimates for the means and covariances. Since these are approximations obtained by taking a sample of the population, the effect of this approximation on the accuracy of the classification must be considered. In theory, the bigger the training sample the less the chance of introducing bias into the estimation and the better the estimate of the statistical parameters of the texture. Consequently the accuracy of the classification should improve as the training area size is increased.

To investigate the effect, various size training areas were taken and the estimates of the statistical parameters made. Both linear and quadratic discriminants were used to assess the effect on these classifiers. A selection of feature subsets were considered.

The four feature subset was chosen as it was best for the quadratic discriminant. The best overall result here is that obtained using the *ad hoc* selection of training area which was used as the basis for all previous results. Finally the best 3 feature linear discriminant subset was chosen as well as the best 2 and 1 feature subsets, the 2 and 1 feature subsets being the same for both types of discriminants. Using these subsets, the

effect of texture on the training area could be measured as well as the effect of the size of the training area, and the number of features.

At this point, a numerical computation aspect is introduced for the calculation of the estimates of the mean vector and the covariance matrix. These are normally calculated from the following formulae :-

The Mean, $$\bar{X} = (\sum_{i=1}^{n} X)/n$$

The Covariance, $$C_{jk} = \frac{1}{n}\left[\sum_{i=1}^{n} X_{ij}X_{ik} - \frac{\sum_{i=1}^{n} X_{ij}\sum_{i=1}^{n} X_{ik}}{n}\right]$$

Where n is the number of samples over which the estimates are formed. This thus requires a squared sum term and a sum squared term both of which may become large. The difference of these two large numbers leads to well known rounding problems. To overcome this problem the following two formulae are presented. They still require only one pass through the data, take only marginally longer to execute than the previous formulae, but are not subject to rounding error even for large numbers. These new formulae use an iterative updating procedure where the new estimate is formed from the previous value.

The formulae are as follows -

The Mean :-

$$\bar{X}_n = \frac{\bar{X}_{n-1} * (n-1) + X_n}{n}$$

The Variance :-

$$S_n = \frac{n-1}{n}\left[S_{n-1} + \frac{1}{n}(X_n - \bar{X}_{n-1})^2\right]$$

The Covariance is found from a similar formula :-

$${}_nC_{jk} = \frac{n-1}{n}\left[{}_{n-1}C_{jk} + \frac{1}{n}({}_jX_n - {}_j\bar{X}_{n-1})({}_kX_n - {}_k\bar{X}_{n-1})\right]$$

The iteration is initialised by the mean being given the first data value and the variance-covariance terms set to zero. (Proofs of these formulae are given in Appendix 2).

For each of the feature sets, results using both the linear and quadratic discriminants were obtained. The texture collage image (figure 4.1) was used as the basis for the test. For each feature set and each discriminant the training area was varied from 10% to 100% in steps of 10% for each of the four textures. Using these training areas and knowing the true classification from the ground truthed image (figure 4.5) the classification errors could be found. The results for the linear discriminant are given in table 4.8 and for the quadratic discriminant in table 4.9. Each table gives the results for

each of the textures. With such information the interaction of texture, discriminant, feature set and training size may be studied. The first general observation is that, as expected, the average classification error decreases as the size of the training area increases no matter which discriminant is used. On closer inspection several anomalies are apparent. Consider the linear discriminant first.

For the seaweed texture, no matter which feature set was used an increasing error was observed with increasing training area size. The other 3 textures have occasional fluctuations but generally give decreasing error with increasing training area size. It is emphasised that the seaweed texture (and its features) is the most non-Gaussian of the set (table 4.1) and the linear discriminant is a most unsuitable model for this texture.

With the quadratic discriminant the situation is more stable, except for the single feature case. With more than one feature, all textures behaved reasonably well showing a decreased error rate with increasing training area size.

The effect of sample size on the classifier was recently studied by Fukunaga and Hayes [4.16] who concluded that the linear discriminant was more robust than the quadratic discriminant for Gaussian samples. It is apparent from the results presented here that departures from normality (in the Gaussian sense) are capable of completely destroying the general concept of increasing sample size producing decreasing error. Furthermore there is evidence to support the claim that the quadratic discriminant produces a greater increase in error reduction for fewer samples as dimensionality increases.

No. Of Features	% Training Area	Seaweed	Cork	Fur	Water	Average
	10	19.09	21.72	13.64	95.79	37.56
	20	26.59	15.50	13.64	94.37	37.52
	30	36.38	10.59	13.64	91.90	38.13
	40	46.50	6.98	13.64	87.22	38.59
1	50	62.11	6.57	15.87	16.86	25.35
	60	60.81	10.18	15.87	16.86	25.93
	70	51.01	6.57	15.87	26.29	24.94
	80	51.01	6.57	15.87	26.29	24.94
	90	51.01	6.57	15.87	26.29	24.94
	100	51.01	6.57	15.87	26.29	24.94
	10	7.00	5.86	12.30	29.12	13.57
	20	6.94	5.60	11.50	23.84	11.97
	30	7.41	5.86	12.07	19.08	11.11
	40	7.59	5.62	12.94	17.18	10.83
2	50	7.24	5.95	13.83	15.97	10.75
	60	7.41	6.56	14.52	14.97	10.87
	70	7.48	6.07	15.07	14.33	10.74
	80	7.80	5.57	15.77	13.05	10.55
	90	8.11	5.50	16.08	11.99	10.42
	100	8.54	5.28	15.50	10.96	10.07
	10	5.78	5.02	7.04	26.46	11.07
	20	5.02	6.00	4.87	20.25	9.04
	30	4.92	7.08	4.63	15.61	8.06
	40	4.84	7.11	5.53	14.46	7.98
3	50	4.82	7.23	6.17	13.20	7.85
	60	5.11	7.42	5.59	11.53	7.41
	70	5.26	7.24	6.02	9.55	7.02
	80	5.54	6.91	6.46	8.14	6.76
	90	6.02	6.64	6.91	6.49	6.52
	100	6.94	6.41	6.37	4.64	6.09
	10	4.78	2.95	5.33	20.24	8.33
	20	5.23	2.74	3.68	15.98	6.91
	30	5.66	3.31	3.50	11.56	6.01
	40	6.05	3.25	3.96	9.14	5.60
4	50	6.47	3.28	4.31	7.46	5.38
	60	7.09	3.13	3.69	5.87	4.95
	70	7.17	2.97	3.77	4.41	4.58
	80	7.78	2.60	3.92	3.52	4.46
	90	8.53	2.29	3.89	1.76	4.12
	100	9.28	2.00	3.45	0.84	3.89

Table 4.8. Results for variable training areas - linear discriminant.

No. Of Features	% Training Area	Seaweed	Cork	Fur	Water	Average
	10	15.85	23.26	9.50	79.35	31.99
	20	23.99	17.05	9.50	72.01	30.64
	30	29.59	17.05	9.50	63.62	29.94
	40	29.59	17.05	9.50	63.62	29.94
1	50	36.85	12.14	9.50	40.64	24.78
	60	49.71	11.18	11.49	26.29	24.67
	70	60.81	11.18	11.49	16.86	25.09
	80	49.71	11.18	11.49	26.29	24.67
	90	49.71	11.18	11.49	26.29	24.67
	100	51.01	8.53	9.50	26.29	23.83
	10	9.67	6.21	8.30	11.58	8.94
	20	8.57	7.84	8.02	13.30	9.43
	30	8.64	7.59	8.17	10.83	8.81
	40	8.52	7.50	7.38	9.60	8.25
2	50	7.48	9.80	6.97	9.86	8.53
	60	7.25	11.02	6.16	9.90	8.58
	70	7.61	9.44	6.19	9.09	8.08
	80	8.00	8.88	5.67	7.82	7.59
	90	8.76	7.93	5.47	6.38	7.13
	100	8.92	7.47	5.22	5.66	6.82
	10	9.72	5.83	6.55	13.85	8.99
	20	9.05	6.08	6.87	12.30	8.57
	30	8.85	6.17	6.14	9.70	7.72
	40	8.26	6.20	4.16	8.49	6.78
3	50	6.52	6.95	2.69	8.74	6.23
	60	6.29	6.95	1.88	8.79	5.98
	70	6.49	6.22	1.63	7.15	5.37
	80	6.86	5.68	1.36	5.61	4.88
	90	7.18	5.12	1.31	4.53	4.53
	100	7.37	4.96	1.23	3.72	4.32
	10	7.94	2.99	10.40	12.25	8.40
	20	7.12	2.98	8.36	11.97	7.61
	30	7.14	3.10	6.71	9.49	6.61
	40	7.00	2.80	5.16	7.54	5.63
4	50	5.26	4.01	3.48	6.61	4.84
	60	4.86	3.95	2.03	6.31	4.29
	70	5.26	3.18	1.96	4.27	3.67
	80	5.65	2.61	1.81	3.50	3.39
	90	5.97	2.08	1.67	3.16	3.22
	100	6.48	1.74	1.56	2.23	3.00

Table 4.9. Results for variable training areas - quadratic discriminant.

With the experimental evidence of a reasonably well behaved quadratic discriminant and decreasing error with increasing training size it was decided to build an adaptive quadratic discriminant. In this, an initial training area would be chosen and as classification proceeded the statistics (mean and covariance matrices) for that class would be updated, using the iterative formulae previously described in equations 4.5 and 4.6. The assumption being made is that more pixels will be correctly classified than not, and thus the updating will provide a more accurate estimation of the statistical parameters as the classification proceeds. The results of implementing such an adaptive quadratic discriminant are given in table 4.10. In this table the results for the same feature sets as were used for table 4.9 are given. In this case the training area was initialised at sizes of 10% to 90% in steps of 10%, and were the same initialised areas as used to obtain the results of table 4.9. As the classification proceeded the discriminant was updated. The timing of this compared with the standard quadratic discriminant is an increase by a factor of only 1.6. The results show improvement at little extra cost. Except for the one feature set, all features gave improved results using the adaptive quadratic. The results are more effective with smaller training areas, as expected. It is noted as this stage that the analysis was concluded after the last pixel was classified, and thus that the pixels classified at the end should be classified more accurately than those at the beginning, it may therefore be advantageous to repeat the classification for those at the beginning. This aspect is investigated further in chapter 5.

Number of Features	% Training Area	Seaweed	Cork	Fur	Water	Average
1	10	22.38	31.03	9.12	93.62	39.04
	20	25.61	24.04	8.39	80.08	34.53
	30	29.59	23.54	8.28	63.24	31.16
	40	29.27	17.05	9.50	63.62	29.86
	50	44.53	12.14	9.66	26.29	23.16
	60	55.06	16.32	10.04	16.86	24.57
	70	60.46	15.00	11.49	16.86	25.95
	80	60.81	11.18	11.49	16.86	25.09
	90	49.80	11.18	11.49	16.96	22.36
2	10	8.97	10.62	9.92	6.69	9.05
	20	8.45	9.59	8.23	6.04	8.08
	30	8.48	7.93	7.26	5.19	7.22
	40	8.43	7.81	7.02	5.36	7.16
	50	7.85	8.99	6.68	6.45	7.49
	60	7.71	9.98	5.96	6.85	7.62
	70	7.94	9.08	6.04	7.03	7.52
	80	8.30	8.68	5.74	6.30	7.26
	90	8.91	8.19	5.78	5.61	7.12
3	10	8.63	8.24	8.64	4.59	7.52
	20	8.47	7.17	6.83	4.11	6.65
	30	8.30	6.72	4.09	3.66	5.69
	40	8.01	6.55	1.97	3.85	5.09
	50	7.08	6.96	1.43	4.30	4.94
	60	6.84	7.25	1.31	4.81	5.05
	70	7.01	6.58	1.30	4.51	4.85
	80	7.34	6.23	1.17	4.19	4.73
	90	7.61	5.77	1.15	3.90	4.61
4	10	7.02	3.45	13.30	4.22	7.00
	20	6.22	2.81	9.13	3.29	5.36
	30	6.32	2.59	5.98	2.80	4.42
	40	6.59	2.34	3.65	2.47	3.76
	50	5.74	3.14	1.95	2.59	3.36
	60	5.50	3.30	1.76	3.05	3.40
	70	5.85	2.90	1.78	2.66	3.29
	80	6.19	2.46	1.72	2.66	3.26
	90	6.45	2.21	1.66	2.62	3.23

Table 4.10. Results for adaptive quadratic discriminant(%).

4.3.5. The "Don't Know" situation.

In this section the idea of a reject or "don't know" option is introduced. This is based on the idea that if a pixel is given a classification with perhaps a 90% probability then this is likely to be acceptable, whereas if the probability was only 50% then this would be likely to be unacceptable. In other words if the estimate of group membership is too low then it is possible to assign that pixel to a "don't know" class. In general this will always improve the error rate by refusing to classify difficult cases. In statistical terms, if $P(G_i \mid \boldsymbol{X})$ is the probability of classifying group i given feature vector $\boldsymbol{X}$ then if group i has the largest probability of membership then it is rejected if $P(G_i \mid \boldsymbol{X}) < t$, where t represents a threshold. Chow [4.17] has shown that:

a) The error rate decreases and the reject rate increases as t increases.

b) The error rate is always less than $t\%$.

c) t is the slope of the error/reject rate trade off curve.

Result (a) implies that improvements can always be made by trading errors for rejects. While (c) implies that there is a law of diminishing returns, i.e. rejecting a small number of cases will give a worthwhile improvement in error rate, but as more cases are rejected the reduction in error rate will not be as great.

As regards image processing some useful results are obtained using the idea of a "don't know" class. Figure 4.9 shows the results of using a threshold of 80% on the classification of the texture collage image of figure 4.1. In this case probabilities of less than 80% were assigned to the "don't know" class, and only these pixels are shown in this image. The classifier was a linear discriminant using 4 features. In this image it is evident that the uncertainty in class membership occurrs primarily at class boundaries, an idea that is intuitive but rarely highlighted in practice.

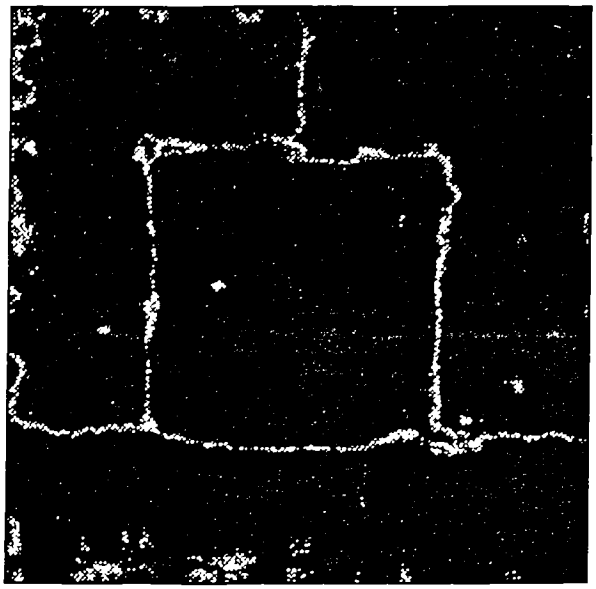


Figure 4.9. "Don't Know" pixels for segmented texture image, 80% threshold.

If figure 4.9 is compared to the ground truth image shown in figure 4.5 the locations of the "don't know" boundaries may be seen to be close to those of the texture changes. This idea of uncertainty in images has been well studied by Granlund and Wilson [4.18] and Wilson and Spann [4.19].

To show the effects of using a "don't know" class, table 4.11 was produced. This shows the results for a linear discriminant on figure 4.1 for varying threshold levels and different numbers of features. As can be seen, in all cases the error rate is improved by increasing the reject rate.

Number of Features	Threshold	Seaweed	Cork	Fur	Water	Don't Know	Average
1	0	38.15	6.31	18.29	40.64	0.00	25.85
	70	9.60	1.60	5.65	9.44	41.00	6.57
	80	7.54	0.30	5.10	3.52	49.12	4.11
4	0	10.21	1.33	4.29	3.74	0.00	4.89
	70	8.12	1.11	3.56	3.12	3.10	3.98
	80	7.09	0.83	2.54	2.59	3.90	3.26

Table 4.11. Effect of classification error rate on reject probability.

Mention should be made at this stage that further improvements in classification can generally be made by employing context, i.e. using the information contained in nearby pixels when assigning a class to a given pixel. No special use of context has been made in this work, except for the use of moderate sized median filters (about 7 * 7). Although the median filter is not generally considered for context classification, it was found useful in this work to help eliminate "noisy" pixels, i.e. those that were surrounded by a large number of pixels of another class, and also aiding interpretation of the boundaries of some of the more complex sonar images analysed. The idea of smoothing the boundaries between textures was suggested by various personnel involved in the interpretation of such sonar images, to make them more like the boundaries that an observer would draw. Figure 4.10 shows the results of a before and after median filter operation on the boundaries of a segmented sonar image.

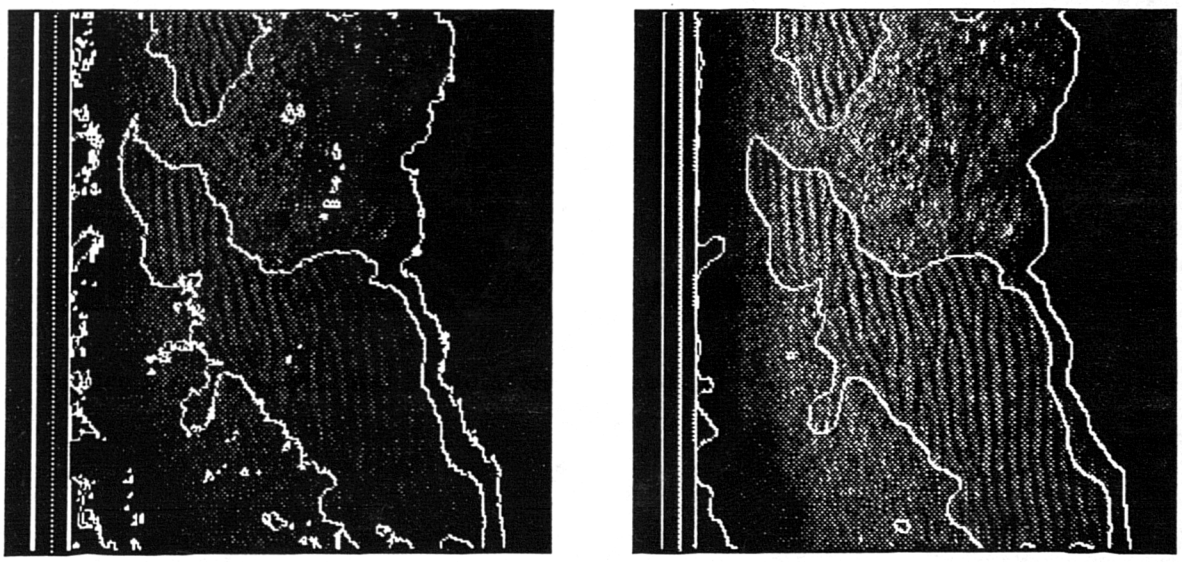


Figure 4.10 Segmentation boundaries, before(left) and after(right) median filtering.

4.4. Some results of segmentation using the operator.

In order to establish further the validity of the operator, using the supervised scheme of this chapter, the results of applying the system to some real images is presented.

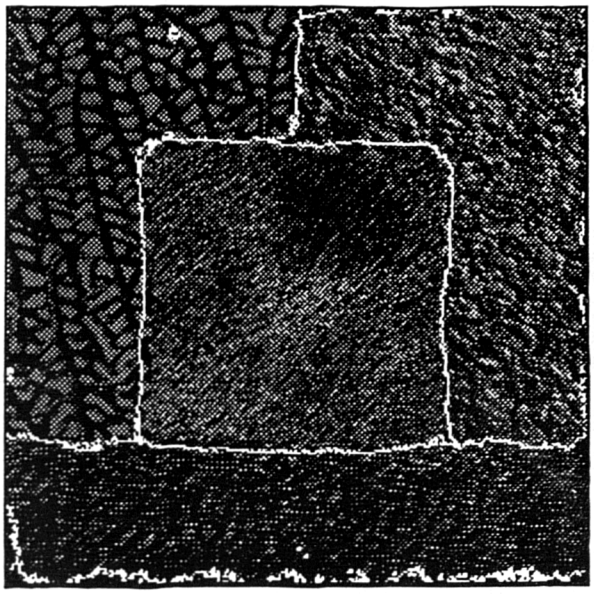


Figure 4.11. Segmented texture collage image.

The segmented image shown in figure 4.11 is the result of applying the system to the texture collage image of figure 4.1. The image size is 512 * 512 with pixel values in the range 0..255, and the boundaries in this artificially composed image are readily seen by the eye. The segmentation was achieved by using directions 1, 2 and 4 of the operator (see figure 2.20), with a level set at 1 and an operator length of 3. One iteration of the operator was run. Following this a low-pass filter in the form of a 21 * 21 averaging kernel was applied to each of the 3 feature images. A supervised scheme was used for the classification and following the classification, an algorithm was applied which found the boundaries between the classes. This boundary image was then overlaid on the original to show the segmentation in figure 4.11. This example clearly shows the ability of the

operator successfully to segment an image with multiple textures. Furthermore, with regard to some of the questions raised at the beginning of the chapter, the segmentation was achieved with only one iteration using directionality. Several textures with a wide range of statistical and spectral properties were separated without knowing the boundary location within the image, and the resolution with regard to boundary accuracy appears adequate. Thus, at least in this image, all the goals of the operator have been achieved. Some further examples are now presented, all of sidescan sonar images, and the same supervised scheme was used on each image. For each image, directions 1,2,4 and 5 were used. Following the generation of the feature images, each was smoothed with a 21 * 21 averaging kernel and subsequently classified into the selected textures using a quadratic discriminant. The boundaries of these classified regions were found and overlaid onto the original images. It is these overlaid boundary images which are shown in figures 4.12, 4.13 and 4.14.

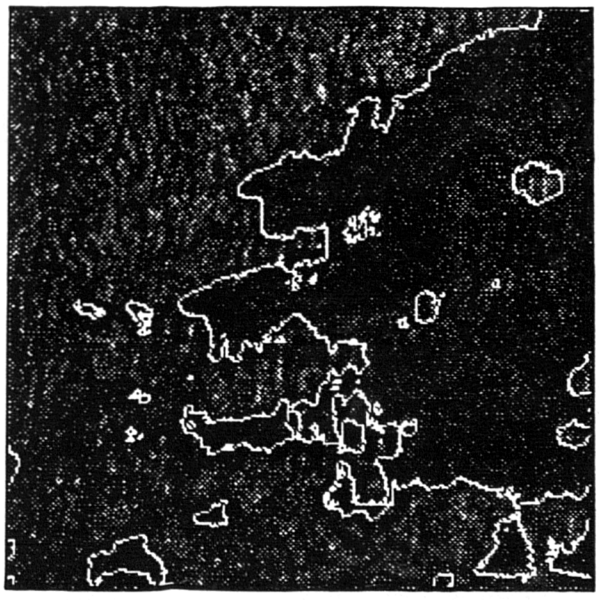


Figure 4.12. Segmented sidescan sonar image.

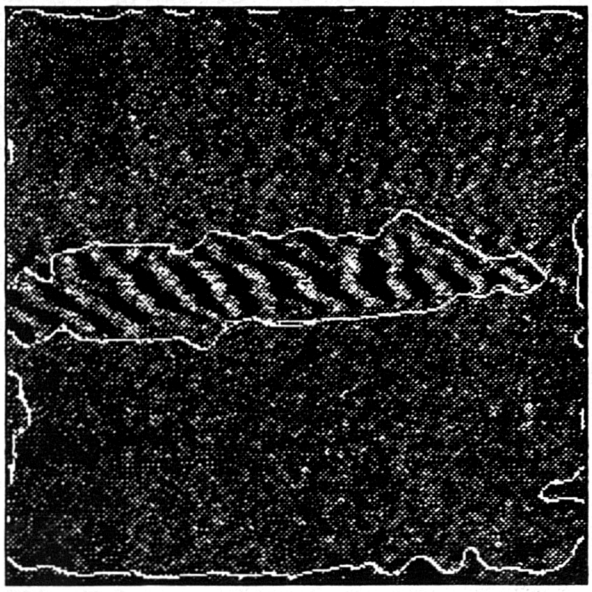


Figure 4.13 Segmented sidescan sonar image.

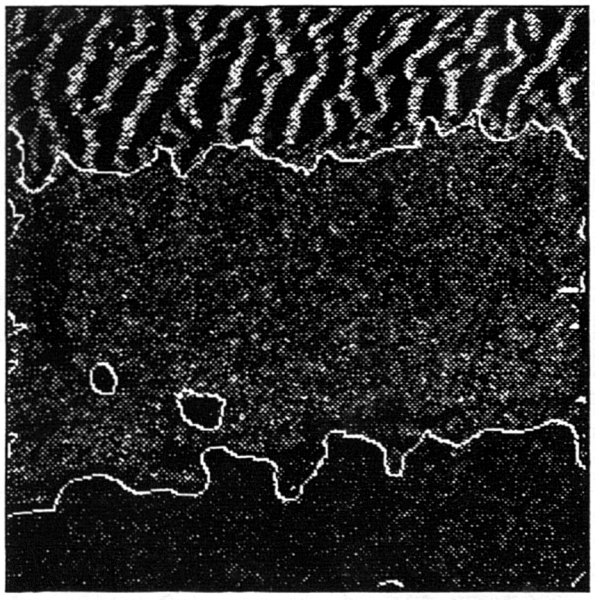


Figure 4.14 Segmented sidescan sonar image.

4.5. Conclusions.

In this chapter some questions regarding the statistical classification of image data have been looked at. Classification systems involving linear and quadratic discriminants were examined. The most common distribution used as the basis for such analyses is the Gaussian. Its applicability to the features of the texture images under consideration was analysed. It was shown that while the feature sets were non-Gaussian, the Gaussian discriminant models used were adequate and provided a most satisfactory vehicle for the final classification. The quadratic discriminant gave slightly better results than the linear, and although it required more processing time than the linear it was to be preferred.

Although relatively few features (7) have been used in the classification, this still required the evaluation of 127 possible feature combinations. It was shown using an exhaustive search, that considerable variation in the quality of the results could be achieved and that it was important to find the subset of features which gave the minimum classification error. Two feature selection algorithms were examined, namely, the branch and bound and the stepwise search. Both produced almost optimum results, the stepwise being preferred for its easier implementation and understanding.

Regarding the problem of training area selection and size, it was shown that the larger the training area then generally the better the classification. From this observation, an adaptive classification system was developed which allowed the mean vector and covariance matrices to be updated. While this system used slightly more processing time, it gave improved accuracy and results.

The idea of incorporating a "don't know" class was discussed. This showed that most uncertainty in the classification system, occurred at the boundaries between different

textures, and that using an extra "don't know" class, gave improved results and greater understanding to the classification system.

Finally, some results of segmentation using the supervised approach were presented, showing that texture boundaries could be found in a wide variety of multi-texture images.

4.6. Authors' papers relevant to chapter 4.

1). L.M.Linnett, A.J.Richardson, *An example of the use of supervised multivariable statistical image analysis*, Research Memorandum, RM/88/5, Heriot-Watt University, Department of Electrical & Electronic Engineering, 1988.

2a). G.T.Russell, G.Shippey, D.M.Lane, L.M.Linnett, A.J.Richardson, *Automated-processing and interpretation of sensory data for deep ocean resource development*, International Ocean Technology Congress, Honolulu, Hawaii, USA, January 22-26, 1989, pp9.7-9.20.

2b). - Also in Ocean Resources Vol II, *Subsea worksystems and Technologies*, chapter 16, pp155-169, Kluwer Academic Publishers, Holland, 1990.

3). L.M.Linnett, G.Shippey, C.Graham, *Subsea image interpretation - A problem in remote sensing*, IEE Colloquium, *Recent developments in image processing: Application in remote sensing*, London, April 1990.

CHAPTER 5

TOWARDS UNSUPERVISED PATTERN RECOGNITION

5.1. Introduction.

The purpose of this chapter is to illustrate the use of some multivariable methods for unsupervised image segmentation. In chapter 4, the analyst chose the training areas which were representative of each texture requiring segmentation. This process alone introduced considerable variation in the results of the classification, although a system was outlined to help overcome this problem. In this chapter, "unsupervised" means that no prior classification has been assigned to any areas of the image, and thus the selection of training areas (and thus what textures are present) will not take place.

As in chapter 4 the purpose of the analysis is to classify regions of an image such that areas of similar characteristics can be identified. In this chapter different textural regions will be classified, the classification result allowing the image to be segmented into these different areas.

This chapter will concentrate on two techniques, cluster analysis, and, in more detail, principal components analysis. These techniques are potentially important in the present scheme of analysis since neither is strictly a statistical technique. No underlying

specific model is required to explain the error structure. In particular, no assumption is made about the probability distribution of the original variables. This is an important factor in this work. Although generally good results were obtained on the basis of the Gaussian model, the data was shown to be non-Gaussian. Thus it is hoped that removing the assumption of a particular probability distribution will give improved results and/or provide more insight into the structure of the features used.

As discussed in chapter 4 certain distinctive features which will distinguish the various component parts of the image are required in order to achieve classification. If the image processing produces features which are not very useful classifiers, then the statistical analysis can do little to improve the situation. The features used are normally those obtained directly from the original feature extraction system, and in practice the number of features can vary from two or three, to more than one hundred [1.53]. As was shown earlier (chapter 4, section 4.3.3) increasing the number of features does not necessarily increase the success of the classification. If a good classification has been achieved using three features it may not be improved by adding a fourth, especially if this feature is correlated to any extent with the other three or is not a good discriminator. The classification would not be improved since the statistical classification would be "diluted" rather than reinforced. This point concerning feature selection was explored in detail in chapter 4, but it will be looked at again in this chapter, but from a different point of view.

5.2. Background to principal component and cluster analysis.

In order to set the scene for the use of these techniques in the image segmentation system under examination some background will be given to each.

5.2.1. Principal component analysis (PCA).

As was mentioned earlier in the introduction, PCA is not a statistical, but a mathematical, technique. In the previous chapter, each variable used in the discriminant for the analysis was given an equal weighting. No one variable was given preference over another, and the problems of choosing which variables to use was discussed and shown to be non trivial. It is the purpose of PCA to reduce the number of variables to find a new set which makes the data easier to interpret and to improve the classification by discarding "irrelevant" information. It should be noted at this point that this is not the same problem of feature selection as in chapter 4. Previously an attempt was made to find the optimum set of variables from the original set. PCA attempts to find a reduced (and hopefully optimum) set of new variables each of which is formed by a suitable linear combination of the original variables. This difference in the two approaches implies that whereas the previous technique hoped to find an optimum set which could always be used in an analysis, in the principal component method *all* the original features must always be generated, but a reduced set of new features will be found from them for use in subsequent analysis. The key element involved in PCA is the calculation of a set of eigenvalues and eigenvectors from a covariance matrix. Since a covariance matrix is always symmetric then the eigenvectors obtained from it will always be orthogonal and real, that is, the eigenvectors will be uncorrelated. (in chapter 4, no mention was made of the effect of correlation between the variables, although in chapter 2 it was shown that between successive iterations of the fractal measure there was a considerable degree of correlation. This aspect will be investigated in section 5.3.) The eigenvectors should thus provide a new set of uncorrelated variables for use in the subsequent statistical analysis.

There are many excellent texts [5.1], [5.2] which discuss the mathematics of

principal components and the derivation of the eigenvalues/eigenvectors. In the light of this only aspects relevant to this work will be considered and a more practical approach will be adopted. In essence the problem of finding the eigenvalues/eigenvectors of a matrix involves the solution of the following characteristic equation -

$$(C - \lambda I).V = 0 \qquad (5.1)$$

C is the covariance matrix, I the identity matrix, V the eigenvectors and λ the eigenvalues. The solution of this equation can be contemplated only by numerical methods when the dimensionality is greater than 2, since ultimately for an n dimensional covariance matrix the roots of an n^{th} order polynomial must be found, as well as the determinant of the matrix. The method used in the present study was the Jacobi method of iterative refinement of a successive series of approximations to the eigenvalues (see program *pcanalim.c* in appendix 3). For an n dimensional covariance matrix the resulting solution produces a set of n eigenvalues, and associated with each eigenvalue is an n length eigenvector. In this study they are presented with the most significant first and those of decreasing significance following in descending order. The eigenvalues produced by the Jacobi method are normalised such that the sum of the eigenvalues is equal to the trace (sum of diagonals) of the original covariance matrix. Since each of the elements of these diagonals represents the variance of each of the original features then the sum of the eigenvalues now represents the total variance of the original data. Thus the first eigenvalue accounts for the greatest variance and the others follow in decreasing order. A convenient geometrical way of illustrating the relationship between eigenvalues and eigenvectors may be obtained as follows:-

Consider the symmetric matrix -

$$C = \begin{bmatrix} 4 & 8 \\ 8 & 4 \end{bmatrix}$$

then $C - \lambda I$ gives

$$\begin{bmatrix} 4-\lambda & 8 \\ 8 & 4-\lambda \end{bmatrix} = 0$$

whose determinant is $(4-\lambda)(4-\lambda) - 8*8$. Setting this to zero produces $\lambda_1 = 12$ and $\lambda_2 = -4$.

Thus $\lambda_1 + \lambda_2 = 8$ = the variance (trace) of the original matrix.

The eigenvalue λ_1 is now substituted back into the characteristic equation (5.1) and this is solved for the eigenvector V_1 for this eigenvalue. Similarly the eigenvector V_2 is obtained. These have values $V_1 = [1,1]^T$, and $V_2 = [-1,1]^T$. By plotting these vectors the result shown in figure 5.1 is obtained.

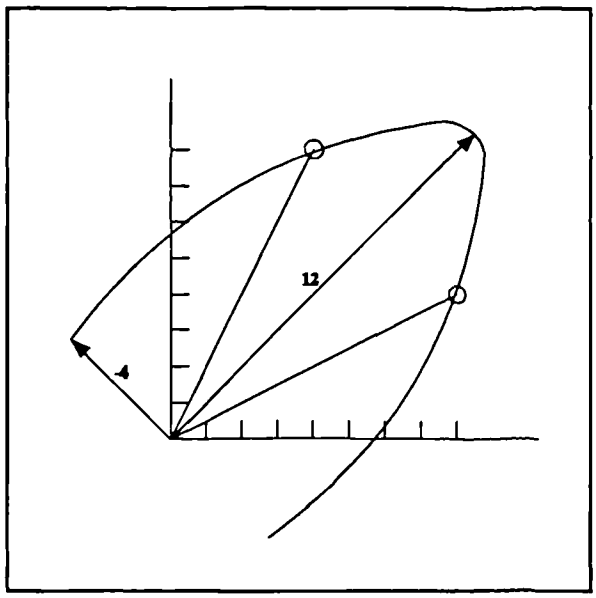


Figure 5.1. Geometric representation of 2D eigenvectors.

In this 2D case, for symmetric matrices only (the only type considered here), the two principal axes are at 45° to the original axes. The first eigenvalue gives the length of the major axis of an ellipse and the second gives the length of the minor axis, the angle of the ellipse being determined by the eigenvectors. Note that if the two rows of the original matrix are equal the ellipse becomes a straight line, the second eigenvalue being zero.

Another aspect to consider in respect of the present image segmentation system is the matter of scaling and its effects on the principal components. Scaling in this context means changing the values of one or more variables by a constant amount. This is often desirable where one variable has much larger values than another and thus would otherwise dominate the analysis. Usually a transformation to zero mean and unit variance is made where all measurements are scaled as -

$$x_{new} = \frac{x_{old} - \mu_i}{\sigma_i}$$

where x_{new} and x_{old} are the values of the transformed and old measurements respectively, and σ_i and μ_i are the standard deviation and mean of the i^{th} variable respectively. This new set of measurements then ensures that all variables are given equal consideration in the subsequent analysis. In the present segmentation system however, all measurements are made in the same manner, variations in value being due to the directionality and properties of the texture, a point which has been deliberately included for reasons outlined in chapter 2. Thus scaling will not be used in this analysis. It is appropriate to mention this as the principal components are not scale invariant. That is, obtaining a result using the covariance matrix will in general not produce the same result as if the correlation matrix is used. Note that if the data is scaled to zero mean (and unit

variance) as outlined above, then the covariance matrix and correlation matrix are the same.

Consider the two dimensional covariance matrix given by -

$$C = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$

where σ_1^2, σ_2^2 are the variances of variables 1 and 2 respectively and ρ is the correlation coefficient between them.

From a consideration of the determinant of this matrix, the eigenvalues and associated eigenvectors may be shown to be -

$$\lambda_1 = \tfrac{1}{2}(\sigma_1^2+\sigma_2^2) + \tfrac{1}{2}\delta, \quad V_1 = \{(\sigma_2^2-\sigma_1^2+\delta), -2\rho\sigma_1\sigma_2\}$$

$$\lambda_2 = \tfrac{1}{2}(\sigma_1^2+\sigma_2^2) - \tfrac{1}{2}\delta, \quad V_2 = \{2\rho\sigma_1\sigma_2, (\sigma_2^2-\sigma_1^2+\delta)\}$$

where $\delta = [(\sigma_1^2-\sigma_2^2)^2 + 4\sigma_1^2\sigma_2^2\rho^2]^{1/2}$

When $\sigma_1/\sigma_2 = 1$, then the ratio of the components of the eigenvectors i.e. $V_{11}/V_{12} = 1$. If however the first variable is multiplied by a factor k then for scale invariance this ratio should be k, however σ_1 now becomes $k\sigma_1$ and the ratio $V_{11}/V_{12} \neq 1$, thus this system is not scale invariant. This may be extended to the n dimensional case. This theory will now be sufficient for the present segmentation system.

5.2.2. Cluster analysis.

In relation to PCA, Cluster analysis has nothing (in its usual sense) to do with variable selection, but is used to find any groupings present in the data. It is not to be confused with Discriminant analysis, where the objective is to classify new objects into existing groups. Cluster analysis attempts to find the groups given the existing objects,

and is a technique suitable for looking at similarities between objects represented by several variables. The method again starts with the original feature data, from which a "distance" matrix is computed using the following formula,

$$d_{ij} = \frac{\sqrt{\sum_{k=1}^{m} (X_{ik} - X_{jk})^2}}{m}$$

where,

d_{ij} = Euclidean Distance.

X_{ik} = Measurement value for object i, variable k.

X_{jk} = Measurement value for object j, variable k.

m = number of variables.

This matrix constitutes a set of distances between objects measured across all the variables. Low values indicate that the two objects are similar, high values indicate dissimilarity.

Many measures of distance have been used, among them the correlation coefficient, and the absolute (or City Block Metric), where

$$d_{ij} = \sum_{k=1}^{m} |X_{ik} - X_{jk}|$$

In fact the Euclidean and City Block metrics are special cases of the Minkowski metrics, defined by,

$$d_{ij} = \left[\sum_{k=1}^{m} |X_{ik} - X_{jk}|^r \right]^{\frac{1}{r}} \tag{5.2}$$

When $r = 1$, the City Block metric is obtained and when $r = 2$, the Euclidean metric is obtained. Another metric used is the Mahalanobis Distance (discussed in a slightly different context in chapter 4), defined as

$$d_{ij} = (X_i - X_j)^T C^{-1} (X_i - X_j)$$

where,

T signifies the transpose of the matrix.

C^{-1} = inverse of the variance-covariance matrix.

X_i, X_j = vectors for objects i,j.

This metric is scale invariant, however if the data is scaled then the Euclidean and Mahalanobis metrics are equivalent. In this work the Euclidean metric is used (it is the most generally used metric). In the program used at present, seven cluster methods are implemented, all being variations produced from the generalised Minkowski formula given in equation (5.2). The available list is given below.

1 = Nearest neighbour or Single linkage.

2 = Furthest neighbour or Complete linkage.

3 = Centroid.

4 = Median.

5 = Group Average.

6 = Gowers' Method.

7 = Wards' Method.

For a more complete discussion of the various methods see [5.3],[5.4]. The method used in the present study was method 1, the nearest neighbour, with the Euclidean metric.

It is worth noting that the Euclidean metric is sensitive to changes in scale, and thus care must be taken in comparing data which has been scaled with that which has not been scaled.

Having obtained the distance matrix, which is symmetrical, it now needs to be sorted to map similar objects together. In this analysis the objects are initially separate, finally they are brought together as one cluster. In Cluster Analysis this is termed a Hierarchical Agglomerative method, as opposed to a Hierarchical Divisive method, which begins with one cluster and divides down until all the objects are separated. As was the case with the metrics, many different methods of analysing the distance matrix have been proposed. The method used in this chapter is termed Nearest Neighbour or Single Linkage Cluster Analysis. The particular computer program used allows seven different methods to analyse the data. (Appendix 3 gives a summary of the computer programs used).

The result of the cluster analysis is a list of numbers showing the distances at

which each object links to another object or group of objects. Similar objects link into the same groups and eventually the groups join together. This data is usually viewed as a dendrogram. (Programs for the production of these are given in Appendix 3).

5.3. Experimental.

In order to show the uses of these techniques within the present image segmentation scheme the image shown in figure 4.1 was used. It has 512 columns, 512 rows and pixel values in the range 0 to 255. The image is made up of four images taken from [1.35]. To the eye, there are clearly four distinct texture regions, and the aim will be to try and distinguish these without the use of training areas, which was the method adopted in chapter 4.

The features used in the following analyses are based on those discussed in chapter 2.

5.3.1. Unsupervised segmentation using two iterations of the operator.

For the results previously shown only one iteration of the operator was used, based on the reasons outlined in chapter 2. However to illustrate the techniques and further to verify the use of only one iteration the first experiment to be described will begin by using two iterations of the operator in all seven directions, which produces fourteen feature images each the same size as the original. From these, each image was broken down into regions of 64 * 64 pixel blocks. The fractal measures for each of the pixels in these blocks were summed to yield a single result for each block, resulting in each of the feature images now occupying 8 * 8 matrices. In statistical terms there is now a classification matrix which has 8 * 8 (64) objects and fourteen variables(features), i.e. a matrix of 64 rows * 14 columns. It is this matrix which is used for the statistical analysis.

Figure 5.2 shows the arrangement of boxes(objects) related to the original image.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

Figure 5.2. Arrangement of boxes in image.

Having obtained this feature matrix, the covariance matrix is calculated, and from this are calculated its eigenvalues and eigenvectors. This yields 14 eigenvalues and 14 eigenvectors. Since the sum of the eigenvalues of the covariance matrix is the trace of the matrix this gives the total variance, each principal axis representing an amount of the original variance proportional to its eigenvalue. To use these eigenvalues and eigenvectors for a Principal Component plot, a transformation is made of the form :-

$$Y_{ik} = \sum_{j=1}^{n} \alpha_{jk} X_{jk} \tag{5.3}$$

where,

Y_{ik} = the "score" for the k^{th} observation on axis i.

α_{jk} = the component of the eigenvector for object k variable j.

X_{jk} = the data value for object k variable j.

In this way each object obtains a score for each eigenvalue.

Eigenvalues	% Contribution	Cumulative Sum
9.292448	77.435	77.435
2.033811	16.948	94.383
0.487683	4.0639	98.447
0.114693	0.9557	99.403
0.040384	0.3365	99.739
0.015541	0.1295	99.869
0.007975	0.0665	99.935
0.005637	0.0470	99.982
0.001036	0.0086	99.991
0.000666	0.0055	99.996
0.000173	0.0014	99.9976
0.000127	0.0012	99.9987
0.000096	0.0008	99.9995
0.000064	0.0005	99.9999

Table 5.1. Eigenvalues for 14 feature problem.

Table 5.1 shows the eigenvalues and the contribution of each vector to the total variance. This shows that the first 2 eigenvectors taken together account for 94.4% of the variance of the original data.

Plotting the scores for each box(object), using the first two eigenvectors as axes (remember that they are orthogonal), produces the result shown in figure 5.3.

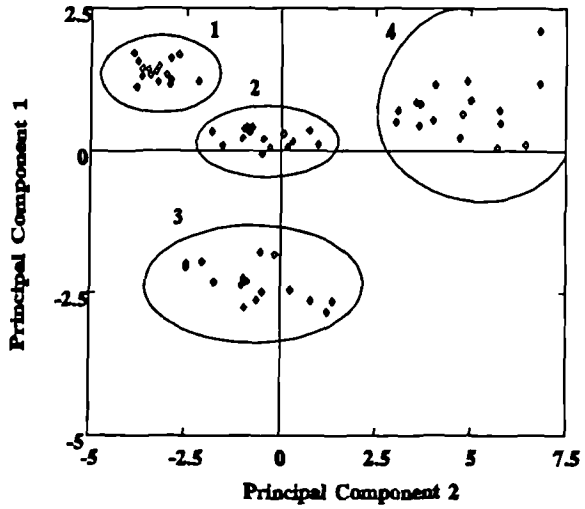


Figure 5.3. Principal component plot, vectors 1,2 - 14 features.

The numbers in the plot represent the textures in the original image (see figure 5.3). As can be seen from this principal component plot, there are 4 groupings. Numbering the groupings as 1,2,3,4 and assigning the boxes in each group to the appropriate number it is demonstrated that the groups represent the different textures of the original image. Figure 5.4 shows how the groupings and boxes map into the original image. The Principal Component plot also shows how well the different classes have been separated by the fractal features. As can be seen there is no ambiguity in assigning objects to the various classes.

1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	3	3	3	3	2	2
1	1	3	3	3	3	2	2
1	1	3	3	3	3	2	2
1	1	3	3	3	3	2	2
4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4

Figure 5.4. Assignment of boxes after classification.

The result of applying this analysis to the 64*14 data set produced a 64*64 distance matrix. Cluster analysis of this produced the output shown as a dendrogram in Figure 5.5.

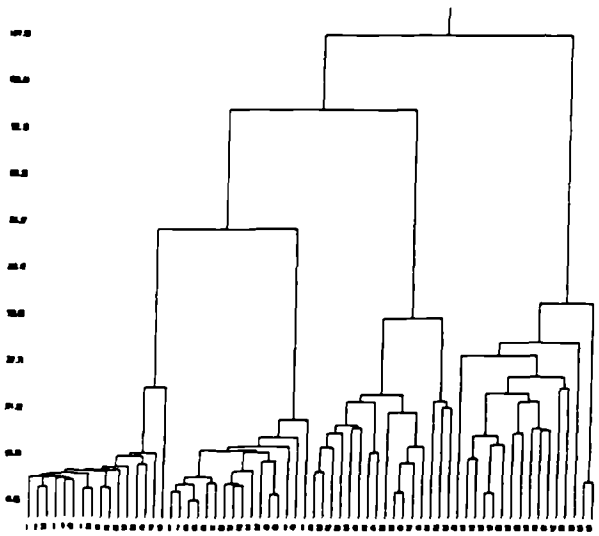


Figure 5.5. Dendrogram, showing clustered groups.

Again, four distinct groupings are seen and if each box is assigned to its particular group in the dendrogram, the same classification results as is shown in figure 5.4.

At this point, although a successful segmentation has resulted there are at least two drawbacks to the system, namely -

a) Two iterations are being used, where only one was used previously.

b) Using the "box" system will produce coarse results, and has only been successful in the present image, since each of the boxes lay on only one texture.

5.3.2. Unsupervised segmentation using one iteration of the operator.

Consider the first point, that of reducing the iterations back to one. In the introduction to principal component analysis, it was shown how the principal components produced an uncorrelated feature set given that there may be correlation among the initial features. It is this fact that enables the principal components to achieve some success, for if the original feature set was uncorrelated then this would be identical to the principal components and there would be little point in performing a principal component analysis. Furthermore, as was shown in chapter 2, there is considerable correlation between the first and second iterations. It may thus be argued that if the principal component analysis eliminates the correlation between the original features reducing the iterations to one will do little to harm the PCA method. There is also correlation between the features within an iteration. This is shown in figure 5.6 where the "box" results for the first iteration only are plotted for directions 5 and 7. Figure 5.8 shows an equivalent plot for directions 1 and 3, illustrating much less correlation for these features. In the next experiment only seven features are used, being one iteration of the operator.

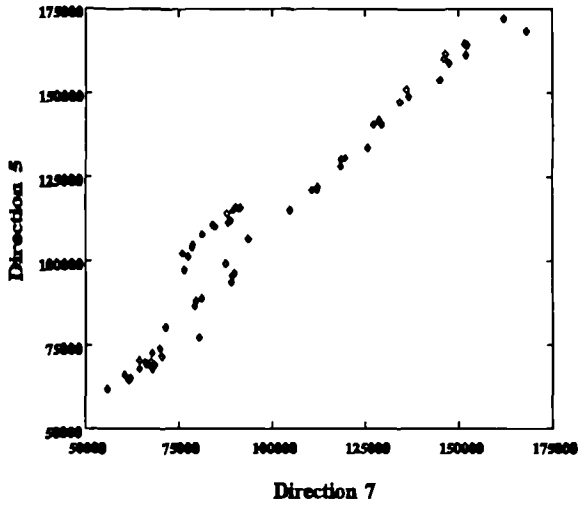


Figure 5.6. Direction 5 measurements plotted against direction 7.

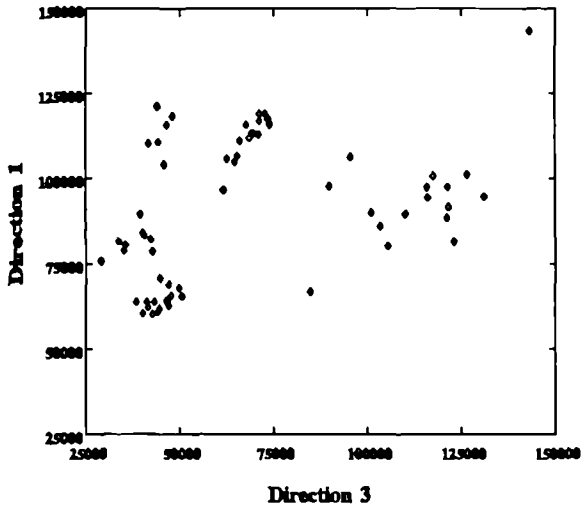


Figure 5.7. Direction 1 measurements plotted against direction 3.

If instead of "boxing" the data values the seven feature images are computed using the seven directions as outlined in chapter 2, and finally each image is smoothed with a 21 * 21 averaging kernel, this produces seven features from which the covariance matrix based on the whole image may be computed. When the principal components are computed for this covariance matrix the eigenvalues shown in table 5.2 are obtained.

Eigenvalues	% Contribution	Cumulative Sum
0.022227	90.775	90.775
0.001460	5.964	96.739
0.000684	2.793	99.531
0.000079	0.324	99.856
0.000023	0.094	99.950
0.000009	0.036	99.985
0.000004	0.015	99.999

Table 5.2. Eigenvalues for 7 feature problem.

The result of plotting the first and third components against each other is shown in figure 5.8. This image will be explained in more detail later, suffice to say at this stage that four "clouds" are visible, and each "cloud" represents a texture within the original image.

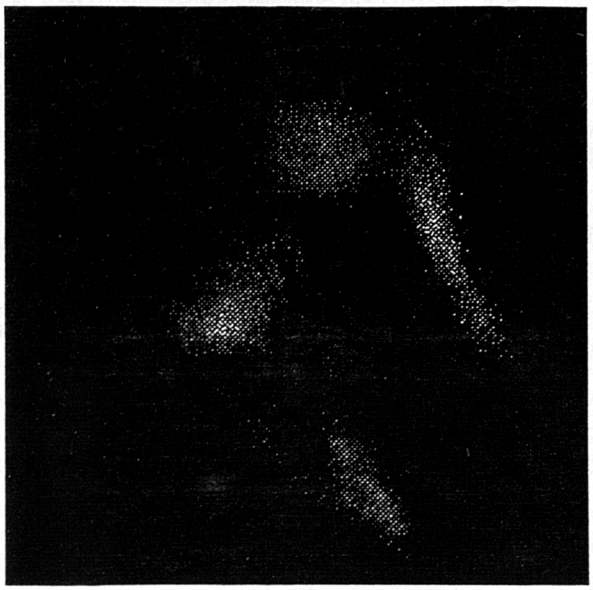


Figure 5.8. Principal component plot, vectors 1,3 - 7 features.

Having obtained a separation of the textures with only one iteration, this helps further to validate the case for using only one iteration which was made in chapter 2. In moving a stage further towards unsupervised recognition, it is worth considering the use of the "new" features, i.e. the principal components, as features in the supervised scheme. From table 5.2 it can be seen that the first three eigenvalues account for 99.5% of the variance or information in the images. Note that several tests have been put forward to find how many components should be used for an analysis [5.5],[5.6]. In this case the first three are chosen, the termination criterion being when an eigenvalue contributes less than 1% to the variance [5.5]. These three components each produce a principal component image. Each pixel in a particular principal component image is made up from the weighted values of the original features and the associated eigenvectors according to equation (5.3). For example, for the image of principal component 1, each pixel is made from the formula -

$I(x,y) = 0.33F_1(x,y) + 0.18F_2(x,y) + 0.35F_3(x,y) + 0.37F_4(x,y) + 0.41F_5(x,y) + 0.42F_6(x,y) + 0.51F_7(x,y)$

where the numeric values are the values of the components of eigenvector 1 and $F_i(x,y)$ is the pixel value at position (x,y) in original feature image i. If these principal component images are used as features with the training image shown in figure 4.4, then a quadratic discriminant classifier produces the segmentation result shown in figure 5.9. The class errors are given in table 5.3 in comparison with the optimum set found using exhaustive search.

Feature Set	Seaweed	Cork	Fur	Water	Average
Principal Component	7.58	3.68	0.63	7.94	4.96
Best Supervised	8.67	0.53	2.77	5.88	4.46

Table 5.3. Principal component and best original quadratic errors.

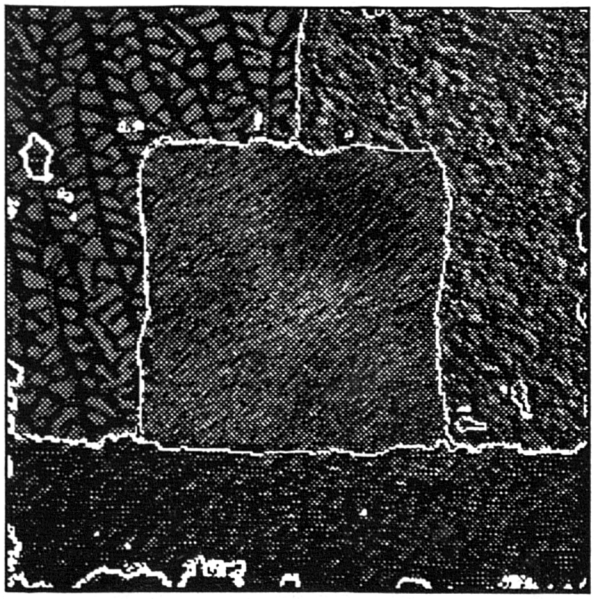


Figure 5.9. Segmentation result using principal component feature set.

This is an encouraging result, since the class errors compare favourably with those for the optimum quadratic found using the exhaustive search. The effort involved in finding the principal component features , however is considerably less than that used in finding the optimal equal weighted feature set. This result has now reached a useful stage, in that a result of comparable quality has been achieved using one iteration of the operator, no "boxing" of pixels and without the need to perform a detailed search for the best feature set. However, as regards unsupervised recognition there is still the problem of eliminating the manual selection of the training areas. In investigating this further it is helpful to consider in more detail the principal components and in particular the principal component plot shown earlier in figure 5.8. This image was formed by considering every pixel in these two principal component images as an (x,y) pair and plotting them with x representing the value of the pixel in principal component image 1, i.e. the row component, and y representing the value of the pixel in principal component image 3, i.e. the column component. Where two sets of coordinates coincide the pixel value in the final plot is incremented, and eventually the whole plot is scaled to the range 0..255 as an image. It is this which is shown in figure 5.8. On inspection it was found that these "clouds" or clusters represented the original textures, and further that they are separated in this principal component space. Note that this representation is preferred to the cluster analysis representation as a dendrogram since it requires much less storage and is much more quickly and easily calculated. The strategy now is to find the centres of these clusters and then to use the pixels around the centres to locate the position of the "training pixels" in the original images. Having then found the training pixels, resort can then be made to the techniques developed for the supervised scheme.

5.3.3. Location of the principal component maxima.

This problem is one of finding the centres of the "clouds" in the principal component plot. Two methods will be discussed, namely the *K*-means algorithm, which has been discussed extensively by Fukunaga [5.7] and the Simplex algorithm of Nelder and Mead [5.8]. The two techniques discussed differ slightly in there goals. Several other methods may be used, e.g. the "isodata" algorithm of Ball and Hall [5.9], which is similar to the *K*-means method, but is more sophisticated in that the spread of the clusters in terms of the standard deviation is required to be specified in assisting to find the cluster centres. Simpler methods, usually using the Euclidean metric, are available and have been well documented in the literature on cluster analysis [5.4].

(a) The *K* - means method.

As a starting parameter this algorithm requires to be told the number of cluster centres to find. With this information the mean position of each of these cluster centres is found using an iterative updating technique. The criteria for finding a cluster centre is based on the minimisation of the sum of the squared distances from all points in a cluster domain to the cluster centre. The algorithm may be conveniently stated as follows -

(1). Initialise the K cluster centres, $z_1(1), z_2(1),...,z_k(1)$ as the first K samples of the data.

(2). At the kth iteration stage, distribute the samples $\{x\}$ among the K clusters using

$$x \in S_j(k) \text{ if } |x - z_j(k)| < |x - z_i(k)|$$

for all $i = 1,2,..,K$, $i \neq j$. $S_j(k)$ is the set of samples with cluster centre $z_j(k)$.

(3). Compute new cluster centres $z_j(k+1)$, $j = 1,2,....,K$ such that the squared

distances from all points in $S_j(k)$ to the new cluster centre is minimised.

The performance index, J_j is

$$J_j = \sum_{x \in S_j(k)} |x - z_j(k+1)|^2, \qquad j = 1,2,.....,K$$

The value which minimises this performance index is the sample mean of $S_j(k)$, and thus the new cluster centre is given by,

$$z_j(k+1) = \frac{1}{N_j} \sum_{x \in S_j(k)} x, \qquad j = 1,2,.....,K$$

where N_j is the number of samples in $S_j(k)$.

(4). If $z_j(k+1) = z_j(k)$ for $j = 1,2,....,K$ then stop, else goto (2).

It is this algorithm which has been implemented in the present study (see appendix 3, program *kmim.c*).

(b) The simplex method.

This method is different from the K-means technique in that it tries to find the maximum (or minimum) of a multidimensional function. It normally will find only one maximum, so it must be restarted at another position in the domain to find the next maximum. It may be conveniently used with the K-means algorithm, where the K-means method finds the approximate maxima for each cluster and then the simplex method finds the centres of each more accurately. A simplex is a geometric figure with $(N+1)$ vertices in N dimensions. The algorithm begins with the coordinates of $(N+1)$ points in the domain and the responses at these points. In this case the response function is the pixel value in the principal component plot. The simplex shape then adjusts itself so that the lowest response coordinate is reflected away from itself to a hopefully higher response. This

reflected point is equidistant from the line joining the previous two higher responses and the original low response point. Many modifications have been made to the original Nelder and Mead algorithm by many authors, allowing the simplex to expand and contract at different rates according to the response function. In this way the simplex moves about the response domain eventually finding a point where a maximum is reached, based on a stopping tolerance. This stopping criterion may be based on the distance moved by the simplex or by the value of the response function.

Using these algorithms the following results were obtained on the principal component plot of figure 5.8. The output of the *K*-means method produced centres at the positions shown in table 5.4.

Texture	PC 1 (x)	PC 3 (y)
Seaweed	63	145
Cork	134	103
Fur	206	156
Water	114	196

Table 5.4. *K*-means centres for clusters of Figure 5.8.

On submission of these values to the simplex algorithm the centres were further updated to produce the results shown in table 5.5.

Texture	PC 1 (x)	PC 3 (y)
Seaweed	62	141
Cork	140	95
Fur	210	164
Water	114	188

Table 5.5. Simplex centres for clusters of Figure 5.8.

With the centres of the texture clusters located, the selection of the training areas may be made. For this, a square of side ten pixels was placed symmetrically around the centre of each cluster. Moving sequentially through the image the location of all pixels within each cluster centre is found. Finally a training image is built up where the pixels either have values dependent on the texture class (or cluster centre), or else they are zero. Table 5.6 shows the numbers of pixels used for each training area and figure 5.10 shows the training image obtained, with the ground truthed texture boundaries overlaid for clarity. From an observation of this image, it is seen that a useful training image has been constructed, but this time without the analyst specifying these areas.

Texture	Number of Training Pixels
Seaweed	3035
Cork	4314
Fur	2315
Water	3359

Table 5.6. Number of training pixels used for each texture.

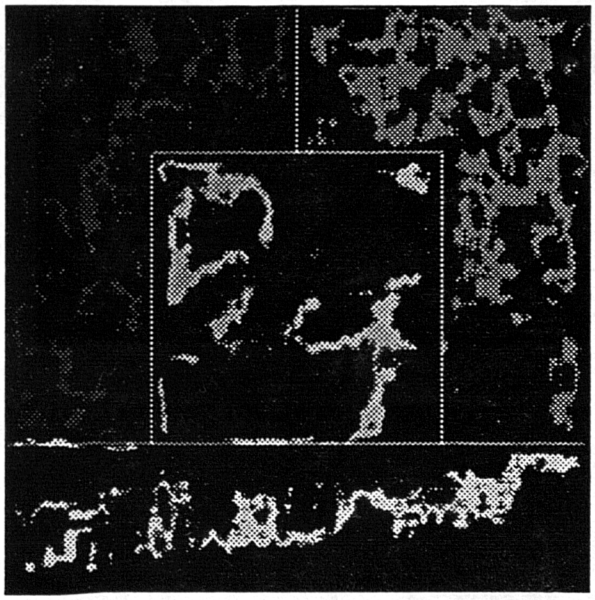


Figure 5.10. Training image for unsupervised recognition.

At this stage all the remaining techniques of the supervised scheme may be used. Using this new training image and the three principal component images as features, the quadratic discriminant is used. However, because of the relatively small numbers of pixels used for the training areas the adaptive quadratic discriminant is used as it was shown to give improved results over the non-adaptive scheme. Using this procedure, the classified image shown in figure 5.11a was produced. Using the final statistics from this image a second iteration of the adaptive quadratic discriminant was applied and the classification result shown in figure 5.11b was obtained. Finally, a third iteration, using as a starting point the statistics from the end of the second iteration, was made and the result shown in figure 5.11c was obtained, with the resulting segmented image shown in figure 5.11d. It is noted that the iterative adaptive quadratic procedure has significantly improved the initial result.

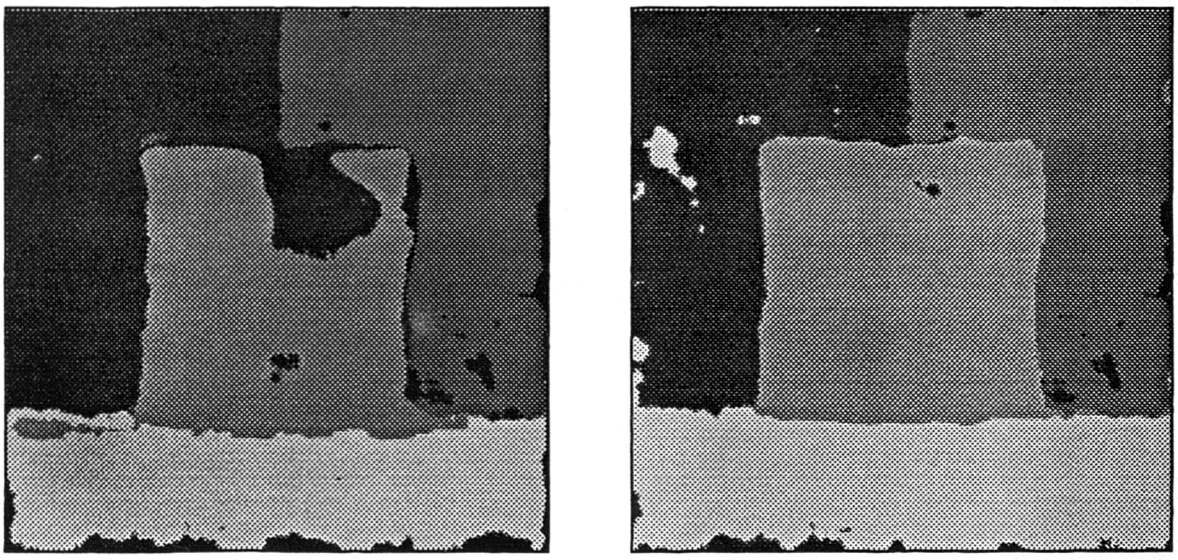


Figure 5.11. Unsupervised classification (a) iteration 1 (left). (b) iteration 2 (right).

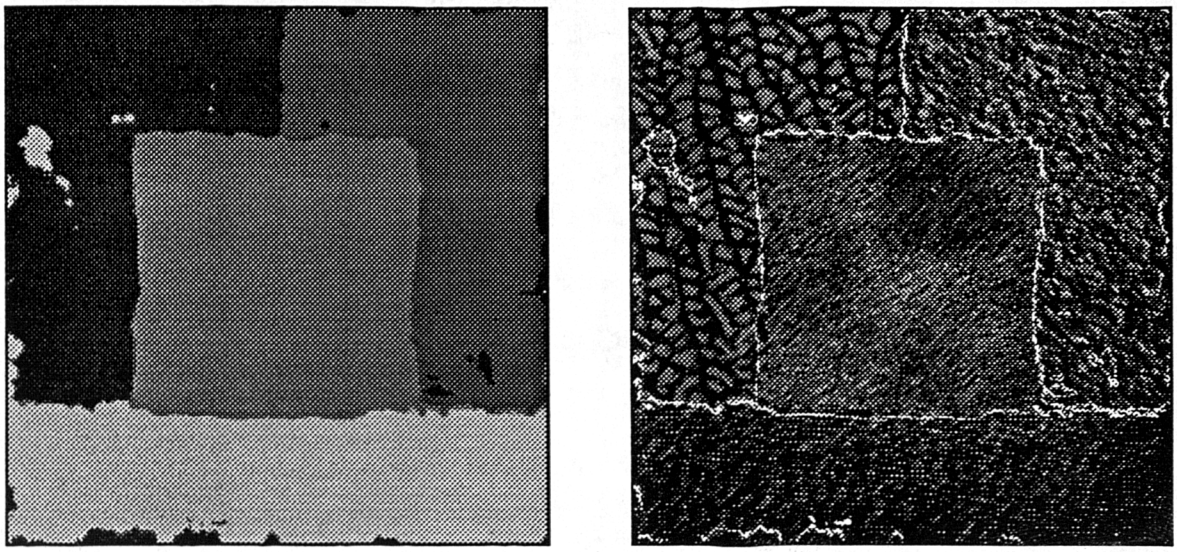


Figure 5.11. Unsupervised classification (c) iteration 3 (left). (d) segmented (right).

In conclusion the unsupervised technique may be summarised as follows -

(1). Obtain the initial seven feature images using one iteration of the operator.

(2). Using these feature images, compute the mean vector and the covariance matrix based on the whole image.

(3). Compute the eigenvalues and eigenvectors of this covariance matrix.

(4). Choose the first n eigenvalues that each give a contribution of greater than 1% to the variance.

(5). Use the corresponding eigenvectors to compute the individual principal component images.

(6). Specify the number of textures and thus find the centres of the respective clusters. Note that the *K*-means and Simplex methods may be applied to an *n* dimensional problem.

(7). From the computed centres, generate the training image.

(8). Use the iterative adaptive quadratic discriminant procedure to produce a classified image.

(9). Produce the segmented image from the classified image.

As a final example, using the scheme outlined above the segmentation of the sidescan sonar image shown in figure 5.12 was produced.

Figure 5.12. Unsupervised segmentation of sidescan sonar image.

5.4. Conclusions.

This chapter has concentrated on methods of unsupervised classification. In this approach, the problems of training area size and selection, and feature selection are avoided. The main method examined was that of principal component analysis. This provided a method for de-correlating the features and providing an optimum feature set formed from a linear combination of the original set.

Using the principal component sets, it was shown that results almost as accurate as that using the best original set could be achieved, this being obtained using the same training areas for comparison. Thus the problems of feature selection could be avoided.

To overcome the selection of training areas, a multi-dimensional principal component space was formed. In this space, principal component "clouds" were formed, each "cloud" representing one of the textures in the original image. Both *K*-means and simplex methods were used to find the centres of these "clouds". Having found the centres, the pixels in a small region around the centre were used to form an unsupervised training set. From this stage, the classification methods of the supervised system were used.

In this way an unsupervised system was produced which gave results comparable to those of the supervised system, but requiring far less processing and operator involvement. The number of textures in the original image still has to be specified. Further work on this problem is continuing.

5.5. Authors' papers relevant to chapter 5.

1). L.M.Linnett, A.J.Richardson, *An example of the use of unsupervised multivariable statistical image analysis*, Research Memorandum, RM/88/4, Heriot-Watt University,

Department of Electrical & Electronic Engineering, 1988.

2). L.M.Linnett, S.J.Clarke, *Sonar image segmentation*, submitted to 7th International Symposium on unmanned untethered submersible technology, University of New Hampshire, 23-25 September, 1991.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

This thesis has examined some of the problems in texture segmentation and synthesis, the application of the work being directed towards an understanding and automated interpretation of video and sidescan sonar signals of the seabed. Several major topics were discussed, namely, an analysis of a texture feature extractor based on fractal ideas, methods of realistic texture synthesis based on fractal ideas, and methods of using extracted features in both the supervised and unsupervised domains to effect a texture segmentation.

Chapter 2 discussed the question of a suitable feature extraction technique that would allow segmentation of a wide range of textures. In examining techniques the idea of a fractal measure was considered. The work follows that of Mandelbrot [1.40] and Peleg [1.44]. In one dimension the feasibility of the method was shown, in particular the ability of the operator, in conjunction with suitable statistical discrimination techniques, to separate signals of different form. When determining the fractal dimension of a signal or image, because of the scaling nature of a fractal, measurements are normally made at many resolutions. This is usually done for two reasons, a) to determine the fractal dimension, and b) to determine the bandwidth over which the signal is fractal.

It was shown that it was unnecessary to use more than one iteration or to determine the actual fractal dimension of the signals in order to separate them. The fact that there was

little spectral change after the first iteration and that subsequent iterations were highly correlated was presented as evidence for using only one iteration. Furthermore the idea of using an operator level greater than zero was discussed and gave improved results when used by statistical classifiers. The kernel size of the operator was found to be optimum at three, although subsequent averaging was necessary for use by the statistical methods. In extending the ideas to two dimensions, the use of directionality was considered, since it was known that this plays a vital role in texture characterisation and analysis. Seven directional features were considered, four using a single direction or two connected, two using two directions or four connected and one using an eight connected set of directions. Using the output of the operator in these directions produced features which, when used by a maximum likelihood discriminant analysis scheme, segmented a wide range of textures. Furthermore, it was shown that the method was suitable for use on a multi-textured image, i.e. one containing several different textures and whose boundaries (and textures) were unknown. An example was presented for the two dimensional, or image, case.

The use of fractals as a model for texture synthesis was studied in chapter 3. It was considered necessary to study the synthesis of textures from the practical point of generating textures for segmentation, but equally important to use the synthesis process to gain a better understanding of the important concepts of the differences between textures. It was demonstrated that several variations on the basic fractal model gave realistic synthetic images. All of the models were generated in the frequency domain and then inverse Fourier transformed to produce the spatial image, and all used random phase, i.e. phase was not taken into account.

The basic fractal model had a centre frequency at DC and the amplitude of the

frequencies rolled off from this peak isotropically in a $1/f^{\beta}$ manner, where the spectral exponent β is related to the fractal dimension. The first variation was to allow the peak frequency to move away from DC, but still have the amplitudes of the frequencies decaying isotropically. This model produced realistic looking textures similar to those of sidescan sonar images. The next variation used this model as a basis but then allowed the amplitudes of the frequencies to roll off in an anisotropic manner. In other words the fractal dimension was allowed to vary in different directions. This too produced realistic image textures. In another variation a band pass fractal model circularly symmetric around DC was used with roll off again isotropic away from DC. This model gave the effect of producing structure in the image. At high band pass frequencies sand like textures were produced, at lower frequencies a seaweed like texture was obtained. In another model, the low pass counterpart to the band pass model, a circular or elliptical region around DC formed the all pass region and the roll off from the cutoff frequency was isotropic. This also produced some interesting textures. Additions were made to some of the models, namely adding more peak frequencies. This did increase the realism but at the expense of more complicated models.

The question of shading and the use of underlying topography of the seafloor on sonar textures was examined. This part of the work showed that when the basic fractal model was used to generate a height field, this could be used to modulate the grey levels of the texture produced by a second model and so produce realistic shading effects. Furthermore the idea of mapping the textures onto the height field for display purposes was shown to be useful for imaging the seabed, as this presented data in a more natural way to those involved in charting the seabed.

The importance of phase in images was illustrated. Some textures are very phase

reliant whereas others are totally independent of phase. The use of the Hilbert transform was explored briefly to show that it is important as a phase detector. Phase quantisation was shown to have little effect on the quality of an image. Only when the number of bits was drastically reduced (to 1) was the image structure destroyed.

The idea of fractal interpolation was explored. This allowed an image to be examined under varying magnifications, without apparent loss of detail. It is felt that this application is perhaps most significant and worthy of further research in the future. It may be argued that, having segmented a multi-textured image, then a small sample may be used as a "seed" to allow that area of texture to be reconstructed with sufficient realism. This is particularly applicable in examining areas of the seabed, for once a sand region has been identified it is not necessary to transmit the exact information for every grain of sand, but only enough to create a realistic likeness. This would have obvious implications for underwater communications and coding. These ideas are being explored further.

In chapter 4 the question of supervised pattern recognition was investigated. It was felt that while this topic may be well covered in statistical texts, some questions related specifically to image analysis should be addressed, and the aim was to bring together some theory and results aimed directly at this area. To this end the general theory of linear and quadratic discriminants for multi-variable analysis was presented. Following on from this some relevant questions not usually tackled in the field of texture segmentation were addressed, namely the normality of the feature data, the influence of the training areas, and the idea of rejecting low probability classes. It was shown that, regarding the non-Gaussian nature of the feature images, the linear and quadratic models based on a Gaussian model were remarkably robust and produced very satisfactory results. It was shown that, despite the results of formal testing of the Gaussian nature of the data

showing it to be quite non-Gaussian, this did not prohibit the application of useful models based on the Gaussian assumption. In fact for the results obtained the tradeoff between a more complex model and the Gaussian for perhaps improved accuracy was not deemed worthwhile for the range of textures examined.

Regarding the question of linear and quadratic discriminants, both models based on the Gaussian distribution were extremely robust with the feature data used. The quadratic form produced slightly better results than the linear one particularly when a texture was very non Gaussian. It was shown that with relatively small numbers of features (four) good segmentation could be achieved using either model, and that this covered a wide variety of texture, both natural and synthetic.

An adaptive classifier was developed to overcome the problems of ad hoc selection of training areas and to provide increased accuracy with only slight increase in execution time. This system, it is felt, would be particularly useful in a database situation which could be updated. A possible extension to the adaptive scheme could be the implementation of an iterative method which allows the classifier to reclassify the image, stopping when the change in classification error reaches a given level.

Regarding the question of feature selection, two standard algorithms were investigated and while neither gave optimum results, due to the non Gaussian nature of the data and the assumption of equal covariance structures, both gave results close to the optimum and certainly useful for practical applications. The characteristic dip in the error/feature curve was noted, but it was felt that because of the low number of features being used (seven), one could do worse than immediately use all seven as the feature set. However, by accurately choosing an optimum or very near optimum set the accuracy could be doubled.

It was shown that by adding a reject option to the classifier, improved results could be achieved and that the majority of all "don't know" situations occurred at the image edges or at the boundaries between textures. It is felt that this aspect could be investigated further to shed more light on the accuracy of boundary estimation in texture segmentation.

As regards extending the work of this section the problems of the non-Gaussian nature of the data should be further addressed, particularly with regard to developing functional discriminants or to developing the classical methods towards models based on neural networks. Since both these ideas are notoriously time consuming, the question of always using an optimum feature set should be addressed. To this end the problems of feature selection in non Gaussian data sets should be investigated.

The use of unsupervised pattern recognition techniques in relation to the segmentation problem was discussed in chapter 5. The emphasis here centred on two techniques, namely, principal component analysis and cluster analysis, where more effort was devoted to the former method.

The correlation of the features was examined and it was shown that some of the features were correlated, and there was also correlation between iterations, which helped reinforce some of the results from chapter 2. It was concluded that only one iteration of the operator was required. The analysis began by considering breaking the image into blocks, and although this would allow coarse segmentation, this was regarded as a weakness in the procedure.

In decorrelating the features the technique of principal component analysis was used. This transformed the original set of seven, possibly correlated, features into a new set of orthogonal and therefore uncorrelated features. Each of these new features was formed from a linear combination of the original features with varying weights. It was

found that employing a subset of these features produced a good segmentation result when used with the supervised techniques. This was a useful result in that no feature selection was used. In extending the technique further, visualisation of a pair of principal components was shown. The K-means and simplex techniques were used to find the centres of the textures or clusters in this principal component space. From these centres, pixels were chosen and related to pixels on the original image, creating a training image which allowed the use of techniques developed in chapter 4. However, instead of using the usual quadratic discriminant an adaptive discriminant was used which continually updated the mean vector and covariance matrices of each class as individual pixels were classified. Using this in an iterative fashion, it was found that successive iterations of this discriminant gave improved results over previous iterations. Good segmentation results. were achieved.

In regard to this chapter, it is felt that future work should be directed at more fully evaluating the features produced by the operator in relation to principal component space and further developing the algorithm for adaptive discrimination to be "more intelligent" in the manner it uses to reclassify pixels. It would also be useful, perhaps, to employ the use of context whereby a more accurate estimate of texture boundaries could be made.

APPENDIX 1.

Table A1a. Exhaustive search - errors for linear discriminant(%).

Feature	Seaweed	Cork	Fur	Water	Average
1	38.15	6.31	18.29	40.64	25.85
2	19.52	20.43	60.55	69.87	42.59
3	10.56	45.32	19.32	64.40	34.90
4	22.84	62.80	15.36	59.70	40.17
5	11.49	25.55	18.62	58.03	28.42
6	24.69	24.39	22.67	59.00	32.69
7	20.34	16.17	20.31	56.96	28.45
12	13.28	3.41	11.73	19.11	11.88
13	12.89	6.91	18.59	5.89	11.07
14	14.27	6.20	19.97	8.82	12.31
15	11.02	4.88	18.05	7.04	10.25
16	11.98	5.08	14.07	13.20	11.08
17	11.04	4.84	16.64	9.52	10.51
23	11.16	9.18	21.39	19.20	15.23
24	19.67	9.04	14.53	27.21	17.61
25	14.96	16.02	8.54	33.91	18.36
26	23.06	16.86	12.55	51.43	25.97
27	20.82	17.33	8.62	52.66	24.86
34	16.65	31.07	12.98	49.59	27.57
35	16.81	7.10	20.16	18.37	15.61
36	14.87	13.61	17.89	6.32	13.17
37	13.48	6.53	20.26	5.57	11.46
45	21.07	4.11	14.68	30.29	17.54
46	30.81	3.71	14.97	34.07	20.89
47	24.22	1.73	11.24	31.86	17.26
56	14.52	13.88	15.14	14.66	14.55
57	12.17	8.07	17.30	6.31	10.96
67	15.27	24.41	3.62	32.29	18.90
123	7.65	5.77	8.29	5.07	6.69
124	7.10	3.12	8.69	10.50	7.36
125	6.68	7.94	7.23	2.66	6.13
126	8.77	3.94	10.30	13.50	9.13
127	7.62	6.59	8.73	6.60	7.39
134	11.80	2.23	15.39	5.06	8.62
135	12.30	4.30	18.14	8.70	10.86
136	14.18	9.69	8.16	2.36	8.60
137	10.17	4.53	12.54	8.10	8.83
145	9.61	1.13	13.05	5.83	7.41
146	9.30	2.92	7.84	9.49	7.39

147	9.24	1.29	9.88	7.94	7.09
156	14.27	8.28	6.20	0.61	7.34
157	10.34	5.81	12.20	4.89	8.31
167	14.02	7.62	6.64	1.09	7.34
234	16.34	9.91	14.57	10.82	12.91
235	13.31	4.82	5.14	17.58	10.21
236	11.74	7.33	11.78	1.55	8.10
237	9.43	4.73	6.35	6.82	6.83
245	20.26	6.41	5.85	12.10	11.16
246	11.37	3.68	12.06	10.74	9.46
247	14.03	1.14	5.93	23.53	11.16
256	13.87	14.75	8.39	1.18	9.55
257	10.22	9.90	7.71	2.67	7.63
267	16.10	21.36	5.11	28.52	17.77
345	17.58	1.62	8.86	11.06	9.78
346	12.58	4.46	10.20	5.79	8.26
347	11.63	1.50	6.07	5.37	6.14
356	16.69	10.10	2.91	4.30	8.50
357	12.33	8.63	15.43	4.21	10.15
367	15.66	7.27	3.15	2.68	7.19
456	15.64	5.56	15.25	14.62	12.77
457	12.61	1.74	9.07	6.01	7.36
467	19.14	3.44	3.66	6.79	8.26
567	15.40	10.56	3.67	3.05	8.17
1234	7.88	3.56	8.09	4.86	6.10
1235	6.51	7.51	6.22	3.03	5.82
1236	10.42	4.80	7.20	2.07	6.12
1237	6.95	5.84	9.11	6.12	7.00
1245	10.59	6.27	5.46	0.41	5.68
1246	7.39	3.00	7.06	9.63	6.77
1247	9.86	2.49	8.26	5.22	6.46
1256	9.39	7.07	5.23	0.59	5.57
1257	6.28	8.18	6.90	1.79	5.79
1267	11.24	6.12	5.48	0.76	5.90
1345	8.32	0.82	13.29	6.94	7.34
1346	10.02	3.28	6.71	1.79	5.45
1347	8.18	0.87	9.67	6.56	6.32
1356	15.25	7.76	5.91	0.63	7.39
1357	9.99	4.39	11.04	5.73	7.79
1367	13.52	5.75	6.60	0.87	6.68
1456	11.20	2.85	5.11	0.60	4.94
1457	9.23	1.12	8.68	3.64	5.67
1467	11.42	2.97	4.97	0.60	4.99
1567	13.75	7.83	5.68	0.62	6.97
2345	17.32	3.28	5.71	6.23	8.14
2346	10.93	4.13	12.07	1.93	7.27
2347	9.20	1.13	5.43	6.37	5.53

2356	11.47	8.32	4.95	1.18	6.48
2357	7.09	6.23	4.91	2.58	5.20
2367	12.32	4.61	3.75	1.58	5.56
2456	13.71	5.20	5.49	0.81	6.30
2457	14.09	4.82	5.29	1.26	6.36
2467	12.75	2.27	4.09	1.84	5.24
2567	14.25	9.99	5.58	1.73	7.89
3456	12.54	3.74	2.71	4.23	5.80
3457	10.21	1.33	4.29	3.74	4.89
3467	12.65	2.35	2.71	2.37	5.02
3567	16.69	7.86	2.66	2.47	7.42
4567	15.66	4.44	3.66	3.01	6.69
12345	11.29	7.56	4.75	0.52	6.03
12346	9.18	3.33	6.18	1.91	5.15
12347	9.02	2.01	8.63	4.70	6.09
12356	9.37	7.10	5.22	0.59	5.57
12357	6.12	7.50	6.83	2.19	5.66
12367	10.51	5.24	5.92	0.71	5.60
12456	10.77	5.55	5.03	0.49	5.46
12457	10.70	5.75	5.83	0.44	5.68
12467	10.89	2.87	4.94	0.68	4.84
12567	10.26	6.50	5.20	0.58	5.64
13456	10.03	2.49	5.33	0.58	4.61
13457	7.32	0.69	8.72	4.31	5.26
13467	10.29	2.59	5.65	0.58	4.78
13567	14.82	6.71	5.69	0.60	6.96
14567	11.72	3.77	4.87	0.61	5.24
23456	11.21	4.00	4.35	0.78	5.09
23457	10.98	3.90	4.70	1.12	5.17
23467	11.02	1.75	3.38	1.03	4.30
23567	11.96	6.00	3.82	1.46	5.81
24567	13.21	4.82	4.97	0.87	5.97
34567	13.30	3.31	2.53	2.40	5.38
123456	10.90	5.27	4.84	0.54	5.39
123457	11.21	6.46	5.33	0.50	5.88
123467	10.07	2.44	5.63	0.61	4.69
123567	10.13	6.68	5.20	0.60	5.65
124567	10.76	5.33	5.17	0.47	5.43
134567	10.66	3.10	5.17	0.59	4.88
234567	11.35	3.65	4.15	0.81	4.99
1234567	11.14	5.48	4.87	0.53	5.50

Table A1b. Exhaustive search - errors for quadratic discriminant(%).

Feature	Seaweed	Cork	Fur	Water	Average
1	38.15	8.53	9.50	40.64	24.20
2	19.52	20.61	84.16	58.09	45.59
3	13.38	30.69	19.32	66.43	32.45
4	22.84	31.02	12.88	78.33	36.27
5	18.42	26.27	13.22	55.69	28.40
6	24.89	28.33	22.67	55.93	32.95
7	24.94	23.34	16.02	51.95	29.06
12	13.15	4.91	5.02	16.62	9.93
13	10.28	5.91	4.18	7.72	7.03
14	12.55	9.09	6.27	8.80	9.18
15	10.32	4.48	5.27	6.57	6.66
16	10.59	6.26	7.77	11.40	9.00
17	10.04	5.05	7.13	8.26	7.62
23	11.80	8.24	5.50	21.89	11.86
24	17.81	7.22	5.94	22.82	13.45
25	15.39	16.98	1.33	20.45	13.53
26	20.90	20.67	2.96	50.57	23.78
27	17.76	24.29	0.51	46.00	22.14
34	14.25	31.89	6.49	38.05	22.67
35	14.77	10.26	9.99	15.22	12.56
36	13.14	20.58	7.99	8.73	12.61
37	10.21	12.27	12.91	7.23	10.66
45	12.54	4.32	9.82	33.25	14.98
46	21.62	3.24	16.65	37.54	19.76
47	17.25	3.91	3.18	27.26	12.90
56	14.04	10.23	7.91	21.33	13.38
57	12.38	13.93	8.29	10.39	11.25
67	16.89	30.62	0.72	27.70	18.98
123	7.70	7.54	0.04	9.88	6.29
124	7.70	3.98	1.05	9.48	5.55
125	8.39	6.29	0.27	7.96	5.73
126	7.64	5.40	1.16	12.29	6.62
127	7.76	7.04	0.31	10.09	6.30
134	7.42	3.69	2.53	9.18	5.70
135	10.48	6.53	3.38	6.96	6.84
136	12.56	7.83	0.15	5.11	6.41
137	10.10	5.59	0.83	8.52	6.26
145	7.27	2.38	2.08	8.51	5.06
146	7.82	3.34	2.44	11.39	6.25
147	6.68	2.29	2.29	10.38	5.41
156	12.21	7.40	0.71	5.73	6.51

157	10.66	4.79	2.66	7.02	6.28
167	13.68	8.97	0.61	4.73	7.00
234	12.55	7.52	4.86	10.17	8.77
235	9.45	6.46	0.31	13.34	7.39
236	11.36	7.79	1.98	6.38	6.88
237	7.90	7.67	0.32	9.59	6.37
245	16.54	9.76	0.19	10.94	9.36
246	10.71	3.93	4.17	10.98	7.45
247	13.65	4.31	0.71	20.27	9.73
256	11.92	10.99	1.39	8.61	8.23
257	9.70	7.56	1.57	9.62	7.11
267	15.64	27.58	0.35	23.75	16.83
345	9.05	3.60	2.46	14.40	7.38
346	10.98	3.41	4.93	10.47	7.45
347	6.64	3.45	2.17	9.06	5.33
356	12.17	8.49	1.37	8.24	7.57
357	11.07	11.21	5.96	7.15	8.85
367	13.35	11.50	0.16	5.88	7.72
456	10.85	0.78	11.36	21.94	11.23
457	10.44	4.22	1.42	11.00	6.77
467	15.95	0.94	1.68	9.39	6.99
567	15.47	9.93	0.47	8.66	8.63
1234	7.24	7.51	0.06	8.67	5.87
1235	7.48	6.00	0.43	7.26	5.29
1236	11.37	7.59	0.04	6.24	6.31
1237	8.30	7.70	0.04	10.35	6.60
1245	12.35	8.87	0.15	5.35	6.68
1246	7.69	4.25	2.31	11.09	6.33
1247	6.74	7.33	0.67	10.72	6.37
1256	9.06	5.64	0.43	5.61	5.19
1257	8.29	4.45	0.92	7.86	5.38
1267	11.66	8.99	0.08	5.30	6.51
1345	7.09	4.46	1.51	8.38	5.36
1346	11.51	1.34	0.71	5.51	4.77
1347	7.60	2.63	0.90	10.06	5.30
1356	13.18	7.99	0.28	5.23	6.67
1357	10.42	5.62	1.21	7.34	6.15
1367	13.67	9.24	0.13	5.48	7.13
1456	8.67	0.53	2.77	5.88	4.46
1457	7.70	1.57	2.17	8.61	5.01
1467	11.74	1.08	1.44	4.73	4.75
1567	14.13	8.05	0.54	5.74	7.12
2345	11.64	9.20	0.16	6.97	6.99
2346	9.37	7.33	3.52	6.64	6.72
2347	6.41	8.01	0.22	9.06	5.92
2356	9.65	6.50	0.43	6.10	5.67
2357	7.83	5.02	0.82	7.58	5.31

2367	11.52	9.31	0.07	6.23	6.78
2456	13.74	1.50	1.47	7.38	6.02
2457	14.45	5.05	0.77	7.84	7.03
2467	12.93	1.11	1.19	5.39	5.16
2567	13.28	8.29	1.39	7.88	7.71
3456	8.83	0.78	3.83	8.93	5.59
3457	7.59	3.78	1.58	8.49	5.36
3467	11.49	1.00	1.57	6.31	5.09
3567	13.61	8.91	0.57	5.99	7.27
4567	13.39	0.65	2.87	9.20	6.53
12345	10.86	8.10	0.26	6.1	6.38
12346	10.99	3.09	0.46	6.2	5.34
12347	7.23	8.08	0.15	9.3	6.35
12356	10.87	5.15	0.47	5.58	5.52
12357	8.10	4.91	0.59	7.79	5.35
12367	12.00	9.07	0.04	6.35	6.86
12456	11.87	1.76	1.45	5.66	5.19
12457	12.01	5.71	0.67	6.39	6.20
12467	11.42	2.38	1.17	5.67	5.16
12567	11.99	5.10	0.64	5.64	5.84
13456	11.21	0.87	1.36	5.35	4.70
13457	7.82	2.54	0.94	8.68	4.99
13467	12.19	1.32	0.70	5.62	4.96
13567	13.74	8.60	0.27	5.64	7.06
14567	12.23	0.63	2.41	5.21	5.12
23456	11.39	1.84	1.25	6.19	5.17
23457	11.19	5.65	0.55	6.56	5.99
23467	10.78	2.77	0.65	6.44	5.16
23567	11.39	5.67	0.59	6.08	5.93
24567	14.30	1.48	1.44	7.38	6.15
34567	11.62	0.71	2.73	6.39	5.36
123456	11.61	1.95	1.00	6.53	5.27
123457	11.24	6.29	0.49	7.05	6.27
123467	11.61	2.92	0.47	7.02	5.50
123567	11.58	5.22	0.52	6.03	5.84
124567	12.77	2.03	1.37	6.20	5.59
134567	11.94	0.96	1.33	5.47	4.93
234567	12.07	1.94	1.15	6.34	5.37
1234567	12.18	2.28	0.99	6.85	5.57

APPENDIX 2

Proofs of iteration formulae for mean and variance.

The mean.

$$\bar{x}_n = \frac{\sum x_n}{n}$$

$$= \frac{\sum x_{n-1} + x_n}{n}$$

$$= \frac{(n-1)}{n}\frac{\sum x_{n-1}}{(n-1)} + \frac{x_n}{n}$$

$$= \frac{(n-1)}{n}\bar{x}_{n-1} + \frac{x_n}{n}$$

$$= \frac{1}{n}\left((n-1)\bar{x}_{n-1} + x_n\right)$$

The variance.

$$S_n = \frac{\sum (x - \bar{x})^2}{n}$$

$$= \frac{1}{n}\left[\sum x^2 - \frac{(\sum x)^2}{n}\right]$$

$$= \frac{1}{n}\left[\sum x_{n-1}^2 + x_n^2 - \frac{(\sum x_{n-1} + x_n)^2}{n}\right]$$

$$= \frac{1}{n}\left[\sum x_{n-1}^2 - \frac{(\sum x_{n-1})^2}{n} + x_n^2 - \frac{x_n^2}{n} - \frac{2x_n \sum x_{n-1}}{n}\right]$$

$$= \frac{1}{n}\left[\sum x_{n-1}^2 - \frac{(\sum x_{n-1})^2}{n} + \frac{n-1}{n}x_n^2 - \frac{2x_n}{n}.\sum x_{n-1}\right]$$

$$= \frac{1}{n}\left[\sum x_{n-1}^2 - \frac{(\sum x_{n-1})^2}{n} + \frac{n-1}{n}.(x_n^2 - 2x_n\bar{x}_{n-1})\right]$$

$$= \frac{1}{n}\left[\sum x_{n-1}^2 - \frac{(\sum x_{n-1})^2}{n} + \frac{n-1}{n}.(x_n - \bar{x}_{n-1})^2 - (\bar{x}_{n-1})^2.\frac{n-1}{n}\right]$$

$$= \frac{1}{n}\left[\sum x_{n-1}^2 - \left(\frac{(\sum x_{n-1})^2}{n} + \left(\frac{\sum x_{n-1}}{n-1}\right)^2.\frac{n-1}{n}\right) + \frac{n-1}{n}(x_n - \bar{x}_{n-1})^2\right]$$

$$= \frac{1}{n}\left[\sum x_{n-1}^2 - \frac{n(\sum x_{n-1})^2}{n(n-1)} + \frac{n-1}{n}.(x_n - \bar{x}_{n-1})^2\right]$$

$$= \frac{1}{n}\left[\left(\sum x_{n-1}^2 - \frac{(\sum x_{n-1})^2}{n-1}\right) + \frac{n-1}{n}.(x_n - \bar{x}_{n-1})^2\right]$$

$$= \frac{1}{n}\left[\frac{n-1}{n-1}\left(\sum x_{n-1}^2 - \frac{(\sum x_{n-1})^2}{n-1}\right) + \frac{n-1}{n}(x_n - \bar{x}_{n-1})^2\right]$$

$$= \frac{n-1}{n}\left[S_{n-1} + \frac{1}{n}(x_n - \bar{x}_{n-1})^2\right]$$

APPENDIX 3.

Relevant computer programs written by the author for this research.

The following gives a brief description of the relevant computer programs used in the course of this thesis. All have been written by the author. All are written in the C programming language, many are also written in Pascal. All except those requiring sophisticated graphics displays will run on IBM PC compatibles, and have been used on a Hewlett-Packard HP9000 series 350 computer. They are listed where possible by the most appropriate chapter.

Chapter 2.

avg2d Will perform a simple moving average on an eight bit image. It requires to be told the names of the input and output images and the kernel size.

addu8 This will add two eight bit images together to produce a third image. It requires the names of two input images and the output image.

cr1dsig This creates a 1D signal. The user specifies the number of samples and the sampling frequency of the signal. Then given the number of frequencies and for each one the frequency, amplitude and phase it computes the output signal.

cr2dsig Is the same as the above except it requires frequencies to be specified for the x and y directions.

frac1d Creates a 1D random fractal signal. It requires the number of samples

(a power of 2) and the fractal dimension. It then computes the signal in the frequency domain and inverse Fourier transforms to the time domain.

gblanket Performs one iteration of the operator on a 1D signal. It requires as input the upper and lower blankets from a previous iteration, or the original signal, also the operator and kernel size. It outputs the upper, lower and difference blankets.

mkchirp Makes a 1D chirp signal given the start and end frequencies and the number of samples in the signal.

normsig Will produce a signal taken from a Gaussian distribution with a specified mean and variance.

newfmu8 Is the 2D equivalent of 'gblanket', performing one iteration of the operator except this time on an eight bit image. This program is the first stage of the processing used in this thesis.

rndsig Is similar to normsig, except the signal is taken from a uniform distribution.

subu8 Subtracts two eight bit images, useful for subtracting upper and lower blanket images.

ttest Performs a sliding *t*-test on a 1D signal. It requires the signal input name and output name and the size of window over which to perform the analysis.

---

chebev Takes as input a 1D discrete signal and then fits a Chebychev polynomial of a specified number of terms to produce a set of output coefficients. This is useful for approximating boundaries in images.

combine This uses a 'jigsaw' boundary image and a number of texture images to create a multi-textured image with the specified boundaries.

crph Creates a 1D signal where the phase has been changed between given start and end points with given start and end values of the phase (in degrees).

fft2dang Computes the power/angle plot representation of an eight bit image. The output is suitable for plotting as a 1D signal.

fft2drad As above, except this computes the power/radius plot.

fftcmplx Computes a fast Fourier transform on a signal. Both forward and inverse transforms are handled as well as real input or complex input signals. It also allows a selection of windowing functions to be applied, or no windowing.

fractinterp Performs a times two (X2) fractal interpolation on an eight bit image. It is used iteratively in a shell file for higher magnifications. It requires a measure of the fractal dimension and the standard deviation of this measure as input.

jigim Takes a blank image and a set of 1D signals. For each signal it prompts the user for the positions of the boundary in the image and then maps the boundary into the image. This jigsaw image is then ready for the

'combine' program.

magph2d Computes a 2D inverse Fourier transform on an image given its magnitude and phase components. This is used in the phase swapping experiments and for combining various phases to the outputs of the various 2D fractal models.

mkanal1d Performs a Hilbert transform on a 1D signal to produce the imaginary, magnitude and phase signals.

mkanal2d As above, except this time for an eight bit input image.

mkfrac1-5 This set of five programs recreates the five models described in chapter 3 for making the various fractal textures. All require varying input depending on the model. The output is an eight bit image and a pseudo 3D plot of its Fourier transform.

my2dfft Performs a 2D fast Fourier transform on an eight bit input image. It also allows the output of the real, imaginary, magnitude or phase images. It can also compute the inverse transform.

spread May be used with the 'combine' program to merge edges of the textures in a combined multi-textured image.

swp2dfft Performs the quadrant swapping operation to produce the output display format for the 2D frequency plot of an image.

texmap Takes two input images, one representing the height field and one representing the texture image, it then maps the two together allowing various options for colour and viewing angle. Only for display on the HP computer system.

unswpft Performs the opposite operation to 'swp2dfft' in preparing swapped

format frequency data for an inverse 2D fast Fourier transform.

wireim Displays in pseudo 3D a wireframe representation of an eight bit image. It is useful for viewing the FFT plots of images. It allows varying viewing angles.

Chapter 4.

allmncov Computes the mean vector and covariance matrix for a whole image using the successive updating algorithms described in chapter 4 and appendix 2.

classim1 Performs a linear discriminant analysis on an eight bit image given the mean vector and covariance matrices and the feature images. The output is a representation of the classified image where each class is assigned a particular grey level.

classim2 Is identical to the above except that it uses a quadratic discriminant.

classx Is identical to 'classim2' except that it allows the user to specify a percentage probability level, below which classes will be identified as 'don't knows'.

combs Computes the number of combinations given a set of features. It outputs all possible configurations.

covtocor Converts a covariance matrix to a correlation matrix.

eqcovar Given a set of covariance matrices this program computes the measure of their equality.

funcdis Performs a functional discriminant analysis for up to three features. In

this a Chebychev approximation is used to represent the multivariate probability density function.

gaussfit Computes the probability that a given input histogram of values is the same as the Gaussian with this mean and variance.

mhobis Measures the Mahalanobis distance between classes for a set of features.

mncovim Is similar to 'allmncov' except that this computes the mean vector and covariance matrices for an eight bit image using the traditional formulae.

segim Creates a boundary image from a classified image. It puts the boundaries at positions where the class (pixel value) changes. The output may be used by 'addu8' with the original image to show the texture boundaries.

stepwise Performs an analysis of the features for each class to give an indication of the suitability of a particular feature for the separation of the classes.

wilks Identical to 'mhobis' except that the Wilks lambda is computed instead of the Mahalanobis distance.

Chapter 5.

adclim2 Performs an adaptive quadratic discriminant classification. As each pixel is classified the means and covariances for that class are updated using the scheme in appendix 2. The output is an updated estimate of these quantities. By repeating the use of the program, an iterative refinement

is obtained.

clust7 Performs a cluster analysis using one of seven possible methods, outlined in chapter 5. The method normally used is method 1, the single linkage method.

distance Computes an Euclidean distance matrix given a set of feature images. This is limited in size to a 128 by 128 image. The output is suitable for use by 'clust7'.

kmim Given a set of features and the number of clusters this computes the positions of the means of these in n-dimensional space. It outputs these positions for each class ready for use by 'simplex' or 'mktrim'.

mk2dhist Produces a 2D eight bit image, where one axis represents one feature and the other axis represents another feature. This is useful for looking at the separation between classes when given principal components.

mktrim Takes the output of 'kmim' or 'simplex' and uses the feature images to create a training image. The user is (optionally) allowed to enter a boxsize which represents the area around the cluster mean from within which training pixels are selected.

pcanalim Produces as output a set of principal component images. It requires the covariance matrix and the original input feature images.

plotdend Uses the output of 'clust7' to produce a dendrogram output graphic.

simplex Given a set of start co-ordinates this program finds the maximum of the data (or function). It is usefully used with the output of 'kmim' for finding the optimum centres of a class.

REFERENCES.

[1.1] A.P.Witkin, *Recovering Surface Shape and Orientation from Texture*, Artificial Intelligence, Vol.17, pp17-45, 1981.

[1.2] A.Gagalowicz, S.D.Ma, *Sequential Synthesis of Natural Textures*, Computer Vision, Graphics and Image Processing, Vol.30, pp289-315, 1985.

[1.3] D.H.Hubel, T.N.Weisel, *Brain Mechanisms of Vision*, Scientific American, Vol.241 No.3, pp150-162, September 1979.

[1.4] B.Julesz, *A Theory of Preattentive Texture Discrimination Based on First-Order Statistics of Textons*, Biol. Cybern., Vol.41, pp131-138, 1981.

[1.5] J.Beck, *Textural Segmentation, Second-Order Statistics, and Textural Elements*, Biol. Cybern., Vol.48, pp125-130, 1983.

[1.6] S.Marcelja, *Mathematical Description of the Responses of Simple Cortical Cells*, J. Opt. Soc. Am., Vol.70 No.11, pp1297-1300, November 1980.

[1.7] D.Gabor, *Theory of Communication*, Journal IEE. London, Vol.93, No.111, pp429-457, 1946.

[1.8] D.A.Pollen, S.F.Ronner, *Visual Cortical Neurons as Localised Spatial Frequency Filters*, IEEE Transactions on Systems, Man and Cybernetics, Vol.SMC-13 No.5, pp907-915, September 1983.

[1.9] J.Daugman, *Two Dimensional Spectral Analysis of Cortical Receptive Field Profiles*, Vision Res., Vol.20, pp847-856, 1980.

[1.10] R.M.Haralick, K.Shanmugam, I.Dinstein, *Textural Features for Image Classification*, IEEE Transactions on Systems Man and Cybernetics, Vol.SMC-3 No.6, pp610-621, 1973.

[1.11] J.S.Weszka, C.R.Dyer, A.Rosenfeld, *A Comparative Study of Texture Measures for Terrain Classification*, IEEE Transactions on Systems Man and Cybernetics, Vol.SMC-6, pp269-285, 1976.

[1.12] M.Unser, *A Fast Texture Classifier Based on Cross Entropy Minimisation*, Proc. 2nd European Signal Processing Conference, Erlagen W.Germany, pp261-264, September 1983.

[1.13] R.W.Conners, C.A.Harlow, *A Theoretical Comparison of Texture Algorithms*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-2 No.3, pp204-222, May 1980.

[1.14] S.W.Zucker, D.Terzopoulos, *Finding Structure in Co-occurrence Matrices for Texture Analysis*, Computer Graphics and Image Processing, Vol.12, pp286-308, 1980.

[1.15] K.Selkainaho, J.Parkinnen, E.Oja, *Structure in Co-occurrence Matrices: Association vs. Agreement*, Proc. 5th Scandinavian Conference on Image Analysis, Stockholm, pp261-268, June 1987.

[1.16] L.S.Davis, M.Clearman, *An Empirical Evaluation of Generalised Co-occurrence Matrices*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-3 No.2, pp214-221, 1981.

[1.17] M.Peitikainen, A.Rosenfeld, *Edge Based Texture Measures*, Proc. 6th International Conference on Pattern Recognition, Munich, pp298-300, 1982.

[1.18] D.Marr, *Vision*, Published by Freeman, San Francisco, 1982.

[1.19] A.Khotanzad, J-Y.Chen, *Unsupervised Segmentation of Textured Images By Edge Detection in Multi-dimensional Features*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.11 No.4, pp414-421, April 1989.

[1.20] O.D.Faugeras, W.K.Pratt, *Decorrelation Methods of Texture Feature Extraction*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-2 No.4, pp323-332, 1980.

[1.21] K.I.Laws, *Texture Energy Measures*, Proc Image Understanding Workshop, pp47-51, November 1979.

[1.22] D.H.Harwood, M.Subbarao, L.S.Davies, *Texture Classification by Local Rank Correlation*, Computer Vision, Graphics and Image Processing, Vol.32, pp404-411, 1985.

[1.23] F.D'Astous, M.E.Jernigan, *Texture Discrimination Based On Detailed Measurements of the Power Spectrum*, Proc. 7th International Conference on Pattern Recognition, Montreal, Canada, pp83-86, July 1984.

[1.24] A.Ikonomopoulos, M.Unser, *A Directional Filtering Approach to Texture Discrimination*, Proc. 7th International Conference on Pattern Recognition, Montreal, Canada, pp87-89, July 1984.

[1.25] H.Knutsson, G.H.Granlund, *Texture Analysis using Two-Dimensional Quadrature Filters*, IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Database Management-CAPAIDM, pp206-213, 1983.

[1.26] G.H.Granlund, J.Arvidsson, H.Knutsson, *GOP, A Paradigm in Hierarchical Processing*, ISMII, 1st IEEE Computer Society International Symposium on Medical Imaging and Image Interpretation, Berlin October 1982.

[1.27] A.V.Oppenheim, J.S.Lim, *The Importance of Phase in Signals*, Proc IEEE Vol 69, pp529-541, May 1981.

[1.28] J.O.Eklundh, *On the Use of Fourier Phase Features For Texture Discrimination*, Computer Graphics and Image Processing, Vol.9, pp199-201, 1979.

1.29] J.G.Daugman, *Six Formal Properties of Two Dimensional Anisotropic Visual Filters*, IEEE Transactions on Systems Man and Cybernetics, Vol.SMC-13 No.5, pp882-887, September 1983.

[1.30] J.Jones, L.Palmer, *An Evaluation of the Two Dimensional Gabor Filter Model of Simple Receptive Fields in Cat Striate Cortex*, J. Neurophysiol, Vol.58, pp1233-1258, 1987.

[1.31] M.Clark, A.C.Bovik, W.S.Geisler, *Texture Segmentation Using Gabor Modulation/Demodulation*, Pattern Recognition Letters, Vol.6, pp261-267, 1987.

[1.32] A.C.Bovik, M.Clark, W.S.Geisler, *Computational Texture Analysis Using Localised Spatial Filtering*, Proc. IEEE Computer Society Workshop on Computer Vision, Miami Beach, Fl., pp201-206, November 1987.

[1.33] P.J.Burt, E.H.Adelson, *The Laplacian Pyramid as a Compact Image Code*, IEEE Transactions on Communications, Vol. COM-31 No.4, pp523-540, April 1983.

[1.34] C.A.Sher, A.Rosenfeld, *Detecting and Extracting Compact Textured Regions Using Pyramids*, Computer Vision, Graphics and Image Processing, Vol.7 No.2, pp129-134, 1989.

[1.35] P.Brodatz, *Textures - A Photographic Album for Artists and Designers*, Dover, New York, 1966.

[1.36] M.Spann, R.Wilson, *A Quad Tree Approach to Image Segmentation Which Combines Statistical and Spatial Information*, Pattern Recognition, Vol.18 No.3/4, pp257-269, 1985.

[1.37] M.Unser, M.Eden, *Multi-resolution Feature Extraction and Selection for Texture Segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.11 No.7, pp717-728, July 1989.

[1.38] H.C.Chen, A.K.C.Wong, *Generalised Texture Representation and Metric*, Computer Vision, Graphics and Image Processing, Vol.23, pp187-206, 1983.

[1.39] A.K.C.Wong, H.C.Chen, *Search Effective Multi-class Texture Classification*, Proc. International Conference on Pattern Recognition and Image Processing, Las Vegas, pp208-213, June 1982.

[1.40] B.B.Mandelbrot, *The Fractal Geometry of Nature*, W. H. Freeman, New York, 1983.

[1.41] R.F.Voss, *Random Fractal Forgeries*, in Fundamental Algorithms For Computer Graphics, R.A.Earnshaw (ed.), Springer-Verlag, Berlin, 1985, pp805-835.

[1.42] P.Burrough, *Fakes, Facsimiles and Facts: Fractal Models of Geophysical Phenomena*, in *Science and Uncertainty*, Proceedings of a Conference, London, March 1984, pp150-169, IBM UK Ltd., Published by Science Reviews Ltd. 1985.

[1.43] A.P.Pentland, *Fractal based description of Natural Scenes*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No.6, pp661-674, November 1984.

[1.44] S.Peleg, J.Naor, R.Hartley, D.Avnir, *Multiple Resolution Texture Analysis and Classification*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No.4, pp518-523, July 1984.

[1.45] H.Nakayama, M.Sone, M.Takagi, *Meteorological Satellite Image Analysis Using Fractal Dimension and Lower Order Statistics*, Proc 5th Scandinavian Conference on Image Analysis, pp261-268, Stockholm, June 1987.

[1.46] G.G.Medioni, Y.Yasumoto, *A Note on Using Fractal Dimension For Segmentation*, Proc of the Workshop on Computer Vision, Annapolis, MD, USA, pp25-30, April-May, 1984.

[1.47] J.M.Keller, R.M.Crownover, R.Y.Chen, *Characteristics of Natural Scenes Related to the Fractal Dimension*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.9 No.5, pp621-627, September 1987.

[1.48] J.M.Keller, S.Chen, R.M.Crownover, *Texture Description and Segmentation through Fractal Geometry*, Computer Vision, Graphics, and Image Processing, Vol.45, pp150-166, 1989.

[1.49] N.Dodd, *Multi-spectral Texture Synthesis Using Fractal Concepts*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.9 No.5, pp703-707, September 1987.

[1.50] A.Fournier, D.Fussell, L.Carpenter, *Computer Rendering of Stochastic Models*, Communications of the ACM, Vol.25 No. 6, pp371-384, June 1982.

[1.51] T.J.Dennis, N.G.Dessipris, *Fractal Modelling in Image Texture Analysis*, IEE Proc., Vol.136 Pt.F No.5, pp227-235, October 1989.

[1.52] A.Gagalowicz, *A New Method for Texture Field Synthesis: Some Applications to the Study of Human Vision*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-3, pp520-532, 1981.

[1.53] D-C.He, L.Wang, J.Guibert, *Texture Discrimination Based on an Optimal Use of Texture Features*, Pattern Recognition, Vol 21, No.2, pp141-146, 1988.

[1.54] S.S.Liu, M.E.Jernigan, *Texture Analysis and Discrimination in Additive Noise*, Computer Vision, Graphics and Image Processing, Vol.49, pp52-67, 1990.

[1.55] W.Siedlecki, J.Sklansky, *A Note on Genetic Algorithms for Large Scale Feature Selection*, Pattern Recognition letters, Vol.10 No.5, pp335-347, November 1989.

[1.56] P.M.Narendra, K.Fukunaga, *A Branch and Bound Algorithm for Feature Subset Selection*, IEEE Transactions on Computers, Vol.26, pp917-922, 1977.

[1.57] J.Y.Hsiao, A.A.Sawchuk, *Supervised Texture Image Segmentation Using Feature Smoothing and Probabilistic Relaxation Techniques*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.11 No.12, pp1279-1292, December 1989.

[1.58] J.Y.Hsiao, A.A.Sawchuk, *Unsupervised Texture Image Segmentation Using Feature Smoothing and Probabilistic Relaxation Techniques*, Computer Vision, Graphics and Image Processing, Vol.48, pp1-21, 1989.

[1.59] S.Peleg, *A New Probabilistic Relaxation Scheme*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-2, pp362-369, 1980.

[2.1] E.L.Hall, *Computer image processing and recognition*, Academic Press, pp38-40, 1979.

[2.2] L.Solymar, *Lectures on Fourier Series*, Oxford University Press, Oxford, 1988.

[2.2] H.-O.Peitgen, P.H.Richter, *The Beauty of Fractals*, Springer-Verlag, Berlin, 1986.

[3.1] J.P.Lewis, *Generalised Stochastic Subdivision*, ACM Transactions on Graphics, Vol 6 No.3, pp167-190, July 1987.

[3.2] G.A.Mastin, P.A.Watterberg, J.F.Mareda, *Fourier Synthesis of Ocean Scenes*, IEEE Computer Graphics and Applications, pp16-23, March 1987.

[3.3] W.J.Pierson, L.Moskowicz, *A Proposed Spatial Form for Fully Developed Wind Seas Based on the Similarity Theory of S.A.Kilaigorodskii*, J. Geophysical Research, pp5181-5190, December 1964.

[3.4] S.Lovejoy, *Area-Perimeter Relation For Rain and Cloud Areas*, Science, Vol.216, pp185-187, 1982.

[3.5] P.Burrough, *Fractal Dimension of Landscapes and Other Environmental Data*, Nature, Vol.294, pp240-242, 1981.

[3.6] K.Perlin, *An Image Synthesiser*, SIGGRAPH Proceedings, Vol 19 No.3, pp287-296, 1985.

[3.7] J.Blinn, *Simulation of Wrinkled Surfaces*, Computer Graphics, Vol 12, No.3, pp286-292, July 1978.

[3.8] P.Kube, A.Pentland, *On the Imaging of Fractal Surfaces*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 10, No.5, pp704-707, September 1988.

[3.9] S.Harrington, *Computer Graphics-A Programming Approach*, McGraw-Hill, London 1987, pp17-20.

[3.10] G.S.Shippey, L.M.Linnett, C.Graham, *Spectral Attenuation Measurements of High Resolution Seismic Records Using Quadrature Band-Pass Filters*, SEG, 60th Annual International Conference, San Francisco, September 1990, pp1075-1078.

[3.11] M.T.Taner, F.Koehler, R.E.Sheriff, *Complex Seismic Trace Analysis*, Geophysics, Vol 44 No.6, pp1041-1063, June 1979.

[4.1] C.Chatfield, A.J.Collins, *Introduction to Multivariate Analysis*, Chapman-Hall, London, 1980.

[4.2] W.W.Cooley, P.R.Lohnes, *Multivariate Data Analysis*, John Wiley and Sons, London, 1971.

[4.3] W.H.Press, B.P.Flannery, S.A.Teukolsky, W.T.Vetterling, *Numerical Recipes - The Art of Scientific Computing*, Cambridge University Press, Cambridge, 1986.

[4.4] K.V.Mardia, *Measures of Multivariate Skewness and Kurtosis with Applications*, Biometrika, Vol.57, pp519-530, 1970.

[4.5] S.P.Smith, A.K.Jain, *A Test to Determine the Multivariate Normality of a Data Set*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 10, No.5 pp757-761, September 1988.

[4.6] J.C.Davis, *Statistics and Data Analysis in Geology*, John Wiley and Sons, New York, pp438-440, 1973.

[4.7] K.V.Mardia, J.T.Kent, B.B.Bibby, *Multivariate Analysis*, Academic Press, London, pp140, 1979.

[4.8] P.A.Devijver, J.Kittler, *Pattern Recognition - A Statistical Approach*, Prentice-Hall, London, 1982.

[4.9] G.Sebestyen, *Decision Making Processes in Pattern Recognition*, Macmillan, New York, 1962.

[4.10] D.J.Hand, *Discrimination and Classification*, Wiley Series in Probability and Mathematical Statistics, 1981.

[4.11] J.W.Van Ness, *On the Effects of Dimension in Discriminant Analysis for Unequal Covariance Populations*, Technometrics, Vol.21, pp119-127, 1979.

[4.12] P.E.Green, *Analysing Multivariate Data*, The Dryden Press, Hinsdale, Illinois, 1978.

[4.13] M.James, *Classification Algorithms*, Collins, London, pp127-148, 1985.

[4.14] C.R.Rao, *Advanced Statistical Methods in Biometric Research*, John Wiley and Sons, New York, 1952.

[4.15] M.S.Bartlett, *Multivariate Analysis*, Journal of the Royal Statistical Society, Series B, Vol.9, pp176-197, 1947.

[4.16] K.Fukunaga, R.R.Hayes, *Effects of Sample Size in Classifier Design*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 11, No 8, pp873-885, August 1989.

[4.17] C.K.Chow, *On Optimum Recognition Error and Reject Tradeoff*, IEEE Transactions on Information Theory, Vol.IT-16, pp41-46, 1970.

[4.19] R.Wilson, M.Spann, *Image Segmentation and Uncertainty*, Research Studies Press - John Wiley, New York, 1988.

[5.1] S.Kaykin, *Modern Filters*, Collier-MacMillan Publishers, London, 1989.

[5.2] G.W.Stewart, *Computer Science and Applied Mathematics*, Academic Press Inc., London, 1973.

[5.3] B.Everitt, *Cluster Analysis*, Published by Heinemann, London, 1977.

[5.4] H.Spath, *Cluster Analysis Algorithms*, Published by Ellis-Horwood, Chichester, 1980.

[5.5] H.F.Kaiser, *The Varimax Criterion for Analytic Rotation in Factor Analysis*, Psychometrika, Vol.23, pp187-200, 1958.

[5.6] R.B.Cattell, *The Scree Test For the Number of Factors*, Multivariate Behav. Res., Vol.1, pp245-276, 1966.

[5.7] K.Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, 1979.

[5.8] J.A.Nelder, R.Mead, *Computer Journal*, Vol 7, pp308, 1965.

[5.9] G.H.Ball, D.J.Hall, *A clustering technique for summarising multivariate data*, Behavl Sci., Vol.12 pp153-155, 1967.